

A MICROCOMPUTER SOLUTION
TO MANEUVERING BOARD PROBLEMS

Kenneth Harper Kerns

NAVAL POSTGRADUATE SCHOOL

Monterey, California



THESIS

A MICROCOMPUTER SOLUTION
TO
MANEUVERING BOARD PROBLEMS

by

Kenneth Harper Kerns

and

Roger Stuart Cooper

Thesis Advisor:

G. A. Kildall

June 1973

7155172

Approved for public release; distribution unlimited.

A Microcomputer Solution
to
Maneuvering Board Problems

by

Kenneth Harper Kerns
Lieutenant Commander, United States Navy
B.A., Vanderbilt University, 1961

and

Roger Stuart Cooper
Lieutenant, United States Naval Reserve
A.B., Princeton University, 1968

Submitted in partial fulfillment of the
requirements for the degree of

MASTER OF SCIENCE IN COMPUTER SCIENCE

from the

NAVAL POSTGRADUATE SCHOOL
June 1973

ABSTRACT

A special-purpose digital computer has been developed using the new MCS LSI microprocessor technology. The primary goal of this work was to solve a fairly complicated computing task using a minimal amount of random logic and limited development time. This computing system solves ships' maneuvering board problems including the determination of course, speed, and closest point of approach of other ships. Ten contacts can be tracked simultaneously. The triangulation involved in these calculations is performed using a decimal version of the CORDIC algorithm. A complete maneuvering situation can be computed in 4.5 seconds. Except for hardware required to drive the displays, all logic is contained in the software-encoded read-only-memory chips, which drive a single CPU chip. Approximately eight hundred man-hours were required to develop the programs and prototype the hardware. The system is modular and easy to maintain, low in bulk, power consumption, and cost.

TABLE OF CONTENTS

I.	INTRCDUCTION	8
II.	THE MANEUVERING BOARD	10
	A. THE EASIC RELATIVE MOVEMENT PROBLEM	10
	1. The Relative Plot	10
	2. The Vector Diagram	11
	B. CTHER USES OF THE MANEUVERING BOARD	13
	1. Stationing Problem	13
	2. Intercept Problem	13
	3. True Wind Problem	13
	4. Military Applications	14
III.	CORDIC COMPUTING TECHNIQUE	15
	A. CCRDIC PRINCIPLES	15
	1. Iteration Equations	16
	2. Convergence Scheme	17
	3. Shifting	18
	4. CORDIC Computing Process	18
	B. APPLYING CORDIC TECHNIQUE TO THE MCS-4	19
	1. Decimal Implementation	20
	2. Rotation Mode	21
	3. Vectoring Mode	22
	C. OTHER IMPLEMENTATIONS CF CORDIC	23
	1. Decimal Approach	23
	2. HF-35 Approach	23
IV.	IMPLEMENTATION	25
	A. HARDWARE CONSIDERATIONS	25
	B. MANEUVERING BOARD SOLUTICN ALGORITHM	26
	1. Solution of the Relative Plot	27
	2. Solution of the Speed Triangle	29
	C. THE CONTROL PROGRAM	29
	D. THE DISPLAY INTERFACE	30
V.	CONCIUSION	32
	A. RESULTS	32

E. FUTURE WORK	35
APPENDIX A USERS' GUIDE TO MANEUVERING BOARD COMPUTER ..	37
APPENDIX E CONTROL PROGRAM LOGIC DESCRIPTION	44
APPENDIX C ESTIMATES OF MANUFACTURING COSTS	50
APPENDIX D FLOW CHART FOR CORDIC ROTATION MCDE	51
APPENDIX E FLOW CHART FOR CORDIC VECTORING MODE	53
APPENDIX F FLOW CHART FOR MANEUVERING BOARD ALGORITHM ..	55
COMPUTER PROGRAM	57
LIST OF REFERENCES	80
INITIAL DISTRIBUTION LIST	81
FORM DD 1473	82

LIST OF TABLES

I.	CORDIC ARCTANGENT CONSTANTS AND SUBITERATIONS	20
II.	CPA PROBLEMS	34

LIST OF DRAWINGS

1. MANEUVERING BOARD SCHEMATIC	12
2. PROTOTYPE KEYBOARD/DISPLAY SCHEMATIC	38

ACKNOWLEDGMENT

The assistance of three Naval Postgraduate School faculty members is gratefully acknowledged. Assistant Professor Gary A. Kildall served as both thesis advisor and chief scrcunger, mysteriously acquiring all sorts of surplus and supposedly defective hardware. Instructor Ray H. Brubaker, second reader and remedial scldering instructor, demcnstrated his infinite capacity for capacitors and small light bulbs in building the LED display and writing its asscciated software. And finally, Assistant Professor Richard H. Franke expended much of his time and more of his patience in premasticating several of the CORDIC papers.

Dei sub numine viget.

I. INTRODUCTION

A rapidly developing and expanding area of computer technology and application is that of microcomputers. Composed of miniaturized software-encoded chips and minimal hardware circuitry, the microcomputer is ideally adapted to specialized tasks and applications. It is the purpose of this thesis to demonstrate the applicability of the comparatively inexpensive microcomputer technology to Navy functions and problems.

An application which has been investigated by the Naval Electronics Laboratory Center, San Diego (NELC), is that of computerizing certain maneuvering board problems, in particular, determining the closest point of approach (CPA) of multiple radar-detected contacts. NELC has constructed a "breadboard version" of its CPA calculator, described in Refs. 1 and 2, making maximum use of off-the-shelf components. The heart of the unit is six MOS ISI chips manufactured by North American Rockwell Microelectronics Company and described in Refs. 3 and 4. They contain a standard array of arithmetic (add, subtract, multiply, divide) and scientific functions and constants (sine, cosine, exponent, natural logarithm, PI). The control program is stored in three Intel programmable read-only-memory (PROM) chips, with four Signetics random-access-memories (RAM) used for data storage. The NELC unit has a twenty key control keyboard and an eleven key (with decimal point) numeric keyboard, while the output is in the form of a light display of six digits plus sign. Information on five contacts can be stored simultaneously. Data input for each contact requires six individual entries unique to that contact, plus two which are the same for all contacts. Five pieces of output information for each

contact are computed and displayed, one at a time as directed by the user, with a total machine computation time for solutions less than sixteen seconds. NELC's method of solution involves manipulation of the input data using primarily the Law of Sines and the Law of Cosines. Inverse trigonometric functions are not available, and they are approximated by multiplicative constants, as indicated by Ref. 5. The NELC CPA calculator has been tested at sea [Ref. 5], and this test showed that a device which provides a quick, accurate, and reliable solution to maneuvering board problems will be eagerly received by potential Navy users.

The system described below has been developed as an alternate microcomputer solution to the CPA problem. The hardware is the Intel MCS-4 microcomputer set. The software functions have been limited to those which are required for the CPA solution. An extensive control program not only orders the various algorithms, but also makes sufficient checks on actions by the user to insure that input is complete and output will be meaningful. The system as implemented uses keyboard input and light display output, although alternate, more automated input/output schemes are suggested.

II. THE MANEUVERING BOARD

When a proceeding ship is near other ships, knowledge of the relationship of its motion to theirs becomes important in preventing collisions at sea. The Hydrographic Office has prepared a plotting sheet (H.O. 2665-10) called the Maneuvering Board which facilitates the solution of a ship's relative movement problem. This solution involves finding the closest point of approach of another ship along with its course and speed. Reference 6 was used as a guide in defining the terms and various plots on the Maneuvering Board.

A. THE BASIC RELATIVE MOVEMENT PROBLEM

Figure 1 illustrates the format of a Maneuvering Board. Ship positions are plotted using the polar co-ordinate system. The center of the plot represents "reference" or "own" ship and any other point represents the position of a "maneuvering" ship, plotted in true bearing and range from own ship at various times. Solutions of relative motion problems with the Maneuvering Board consist of two distinct but related parts: the relative plot and the vector diagram, the latter often called the speed triangle.

1. The Relative Plot

In the relative plot the "maneuvering" ship is plotted at various times: its position at the beginning of the plot is labeled M1, and subsequent positions are labeled M2, M3, etc., as indicated in Figure 1. The line through M1, M2, M3, etc., is called the relative motion line, defining the direction of relative motion and the distance of relative motion. From this plot are obtained the bearing

and range of the maneuvering ship from own ship at any time and the bearing and range of the closest point of approach of the maneuvering ship from own ship.

The closest point of approach is indicated as the point labeled CPA in Figure 1. The range of CPA is simply the distance from the center of the plot to this point. The bearing of CPA is equal to the direction of relative motion plus or minus ninety degrees, dependent upon the quadrant containing the CPA point. The time of CPA is obtained by adding to the time of M1 the quotient of the measured distance from M1 to the CPA point and the speed of relative motion. This last computation can be accomplished through use of the logarithmic time, speed, and distance scale nomogram found at the bottom of the Maneuvering Board (H.O. 2665-10) .

2. The Vector Diagram

The vector diagram is used to solve for the course and speed of the maneuvering ship. The vector diagram is a triangle composed of the three vectors shown in Figure 1. Vector er indicates the course (vector direction) and speed (vector magnitude) of own ship, while Vector em represents the course and speed of the maneuvering ship. Vector rm represents the direction and speed of relative motion. Thus, given own ship's course and speed and the direction and speed of relative motion determined from the relative plot, the course and speed of the maneuvering ship is found by drawing vector em to complete the triangle erm .

MANEUVERING BOARD SCHEMATIC

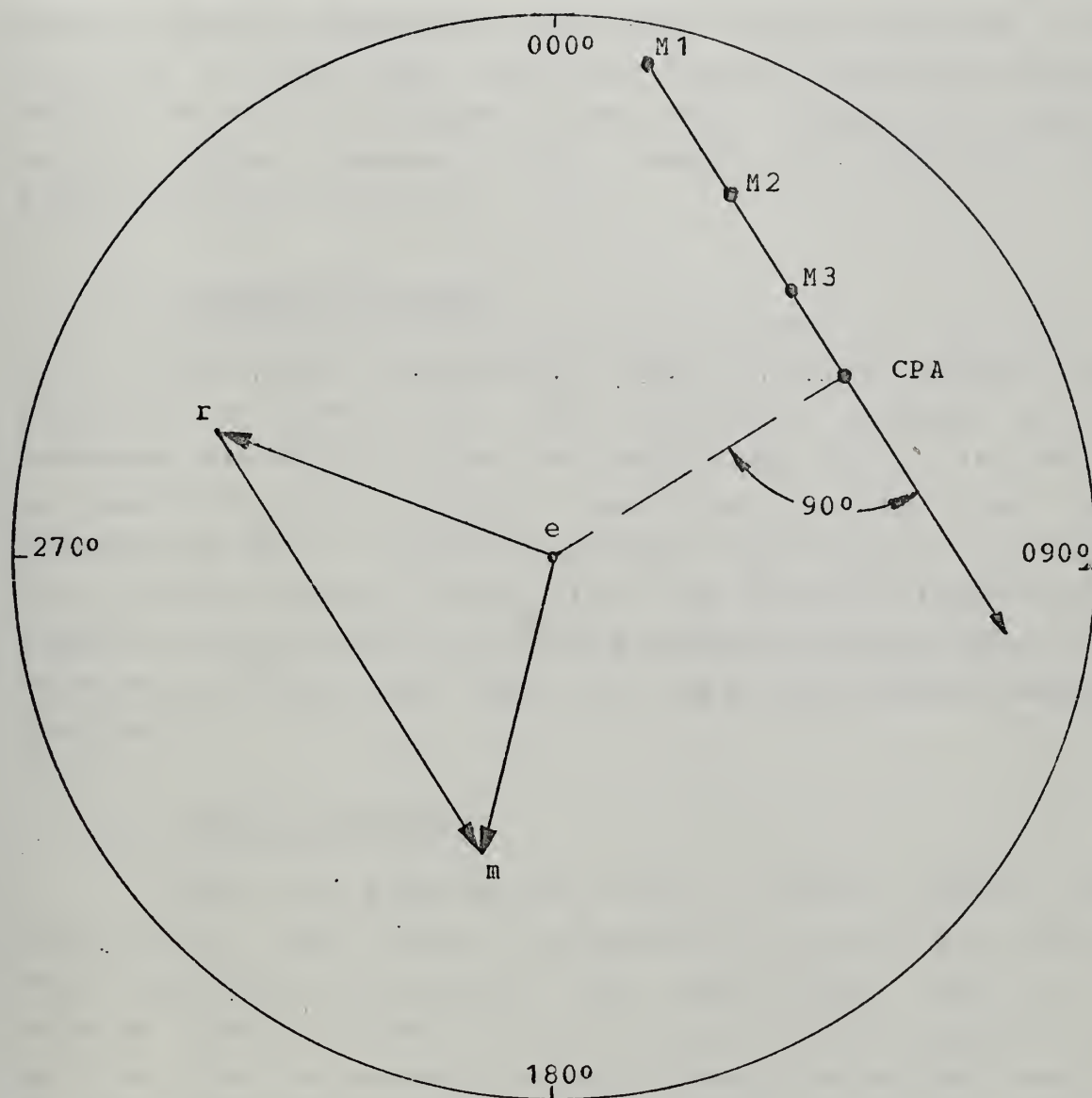


FIGURE 1

B. OTHER USES OF THE MANEUVERING BOARD

All maneuvering board problems utilize the basic relative motion problem discussed above. In addition to determining CPA information, the Maneuvering Board can also be used to find the required course and speed to take station on or to intercept another ship, to find true wind, or to find courses and speeds for scouting and torpedo-firing situations.

1. Stationing Problem

Given the maneuvering ship's course, speed, and position at some time, the stationing problem is to determine own ship's course, or course and speed, in order to maneuver to a certain range and bearing from the maneuvering ship. The basic solution technique is to draw the relative motion line from the starting maneuvering position to the point of required range and bearing from the maneuvering ship, and then to solve the relative motion problem.

2. Intercept Problem

Given the maneuvering ship's course, speed, and position at some time, the problem is to find own ship's course and speed to intercept the maneuvering ship by a certain time. To find a solution, the relative motion line is drawn from the maneuvering ship's position to the center of the maneuvering board, after which the relative motion problem is solved.

3. True Wind Problem

Since the known apparent wind is the resultant of ship's motion and true wind, the maneuvering board can be

used to solve for true wind. The solution for true wind is simply the \mathbf{ew} vector of the vector diagram, where ship's motion is the \mathbf{er} vector and the measured apparent wind is the \mathbf{rw} vector.

4. Military Applications

The Maneuvering Board is used for other problems such as scouting and torpedo problems. These problems involve determination of courses and speeds to undertake a desired scouting or search path, or a course and speed required to shoot a spread of torpedoes. Dutton [Ref. 6] provides descriptions and solution techniques for these problems as well as the problems outlined above.

III. CORDIC COMPUTING TECHNIQUE

The maneuvering board algorithm described below is based upon a vector analysis of the data, which in turn requires certain transcendental functions: sine, cosine, inverse tangent, and modulus. Sine and cosine generating techniques had already been implemented on the MCS-4, as described in Ref. 7, with average run times of 650 and 750 milliseconds, respectively. It was recognized that these routines would yield an unexceptable total computation time if used in the maneuvering board algorithm. Consequently, the CORDIC algorithm described below was investigated and eventually implemented, significantly reducing the time required for each function.

The COREIC (CO-ordinate Rotation DIGital Computer) technique was developed by J. E. Volder in 1959 [Ref. 8]. It is a simple, accurate, and relatively fast method for generating transcendental functions involving only the operations of addition, subtraction, and shifting.

The principle involved in CORDIC is to rotate an imaginary vector represented by its x,y components through a sequence of angular steps of predetermined magnitude. At each step the values of the angle and the x,y components of the vector are modified according to a set of equations that can be implemented with add, subtract, and shift operators.

A. CORDIC PRINCIPLES

J. S. Walther has described [Ref. 9] a single unified algorithm for the calculation of elementary functions using co-ordinate rotation in a linear, circular, or hyperbolic co-ordinate system. For the sake of simplicity, the description of the CORDIC technique is based upon the

circular co-ordinate system.

1. Iteration Equations

Consider a vector $P = (x, y)$ with radius $R = (x^2 + y^2)^{1/2}$ and angle $A = \tan^{-1} y/x$. Let a new vector

$P_{i+1} = (x_{i+1}, y_{i+1})$ be obtained from $P_i = (x_i, y_i)$ by rotating the co-ordinates by $\tan^{-1} D_i$ according to

$$x_{i+1} = x_i \pm y_i D_i \quad (1)$$

$$y_{i+1} = y_i \mp x_i D_i \quad (2)$$

The (\pm) and (\mp) signs indicate that $y_i D_i$ and $x_i D_i$ are either added to or subtracted from the first term depending on the polarity of $\tan^{-1} D_i$, in which D_i is an arbitrary value that will be described below.

The angle and radius of the new vector P_{i+1} in terms of the old vector P_i are

$$A_{i+1} = A_i - \tan^{-1} D_i$$

$$R_{i+1} = R_i (1 + D_i^2)^{1/2},$$

where $(1 + D_i^2)^{1/2}$ represents the elongation factor of the magnitude of the vector after each co-ordinate rotation.

For n iterations:

$$A_n = A_0 - a$$

$$R_n = R_0 K,$$

where

$$a = \sum_{i=0}^{n-1} \tan^{-1} D_i$$

and

$$K = \prod_{i=0}^{n-1} (1 + D_i^2)^{1/2}.$$

Let a third variable Z be provided for the accumulation of the angular variations $\tan^{-1} D_i$ so that

$$Z_{i+1} = Z_i + \tan^{-1} D_i. \quad (3)$$

Solving the set of equations (1), (2), and (3) for n iterations, the final values are

$$X_n = K(X_0 \cos a + Y_0 \sin a)$$

$$Y_n = K(Y_0 \cos a - X_0 \sin a)$$

$$Z_n = Z_0 + a.$$

2. Convergence Scheme

The angle A of vector P can be forced to zero by a converging sequence of rotations, $\tan^{-1} D_i$, which at each step brings the vector closer to the positive x -axis. The direction of rotation must be determined at each step such

that

$$|A_{i+1}| = | |A_i| - \tan^{-1} D_i |. \quad (4)$$

The same idea applies to forcing the angle in Z to zero. It can be shown that A or Z converges to within $\tan^{-1} D_{n-1}$ of zero in n iterations [ref. 9].

3. Shifting

The CORDIC technique requires the use of shifting to effect the multiplication in equations (1) and (2). Let $D_i = r^{-i}$ where r is the radix of a number system and i is any positive integer; then a multiplication of x by D_i is a shift of x by i places to the left. The integer i must be chosen such that the angle $\tan^{-1} r^{-i}$ satisfies the convergence criterion discussed above. Note that the accuracy at the n -th iteration is approximately equal to $n-1$ places [ref. 9].

4. CORDIC Computing Process

In the computing process iteration equations (1), (2), and (3) are used, while the direction of rotation is determined by equation (4). The $\tan^{-1} D_i$ constants are preloaded in either degrees or radians for $i=0,1,2,3,\dots,n$.

There are two computing modes, rotation and vectoring. In the rotation mode Z is driven to zero, and in the vectoring mode A is driven to zero; that is, $Y_n = 0$.

With the circular co-ordinate system as a basis, entering $X = 1/K$, , $Z = \text{some angle } \theta$, and using the rotational mode, the results will be

$$X = \cos \theta$$

$$Y = \sin \theta$$

$$Z = 0,$$

which are accurate to approximately $n-1$ places. In the vectoring mode entering $X = x_0$, $Y = y_0$, $Z = 0$ leads to

$$X = K (x_0^2 + y_0^2)^{1/2}$$

$$Y = 0$$

$$Z = \tan^{-1} y_0/x_0,$$

accurate to approximately $n-1$ places.

Using the linear or hyperbolic systems as a basis to the CORDIC technique, multiplication, division, square root, hyperbolic sine and cosine, or inverse hyperbolic tangent functions are obtained. The remaining transcendental functions can be computed from mathematical identities involving the CORDIC-derived functions. Reference 9 supplies more information in this area.

B. APPLYING CORDIC TECHNIQUE TO THE MCS-4

In order to solve the basic relative movement problem of the maneuvering board, conversions between the polar and rectangular co-ordinate systems are required. The CORDIC technique is ideally suited for this task since it provides the sine and cosine values of an angle in the rotation mode and $(x^2 + y^2)^{1/2}$ and $\tan^{-1} y/x$ in the vector mode.

1. Decimal Implementation

The MCS-4 prototype uses the fixed point decimal number system. The CORDIC technique was developed in fixed point decimal form to facilitate the required shifting operation, since in this case shifting is, in effect, a relocation of the decimal point.

The radix of the decimal system is ten, thus D_i was chosen as 10^{-i} and angular steps of $\tan^{-1} 10^{-i}$ are used. In order to obtain a desired four digit accuracy, five angular steps were required. The values of $\tan^{-1} 10^{-i}$ decrease by approximately 1/10 whenever i is increased by one.

To cover all angles between zero and ninety degrees, several subiterations were required for each angular step, as outlined in Table I.

TABLE I

CORDIC ARCTANGENT CONSTANTS AND SUBITERATIONS

i	$\tan^{-1} 10^{-i}$	Number of Subiterations
0	45°	1
1	5.7105931°	7
2	0.5729386°	9
3	0.0572957°	9
4	0.0057295°	10

The number of subiterations j for each angular iteration i was determined from the inequality

$$\sum_j \tan^{-1} 10^{-(i+1)} \leq \tan^{-1} 10^{-i}.$$

This was done to minimize the total number of iterations, thus reducing the execution time of the MCS-4 prototype.

For any angular step numbered four or greater, the number of subiterations will be ten. Values of $\tan^{-1} 10^{-i}$ were preloaded into ROM in degrees. It should be pointed out that angular values could have been represented in radian units, but degrees were chosen to allow the user to enter information without the need for conversion.

2. Rotation Mode

In the rotation mode the sine and cosine of an input angle ranging from zero to 360 degrees is computed. The flow chart for this process is shown in Appendix D.

The initial values are:

$$X = 1/K = 0.682594516$$

$$Y = 0$$

$$Z = 0.$$

K represents the elongation factor which is precomputed from the formula

$$K = \prod_{i=0}^4 (1 + 10^{-i j/2}),$$

where j represents the number of subiterations for each i as in Table I above.

Prior to the first angular iteration of forty-five degrees, the input angle is tested for polarity and the x, y components rotated plus or minus ninety degrees to place the angle in the first or fourth quadrant. This is done so that

at the next rotation of plus or minus forty-five degrees, the angle will be within forty-five degrees of the positive x-axis.

At each angular iteration, Z is tested and driven to zero; that is, if Z is positive, the rotation is done in the negative direction, or if negative, rotated in the positive direction. The iteration equations are then carried out by either addition or subtraction, and by a shift of i decimal places to the left. After the final iteration

$$\begin{aligned} X &= \cos Z \\ Y &= \sin Z \\ Z &= 0. \end{aligned}$$

3. Vectoring Mode

In the vectoring mode the y component of the pseudo vector is tested and driven to zero. This test is equivalent to driving the vector angle A to zero. The flow chart for this process is shown in Appendix E. The appropriate values are entered for X and Y with Z equal to zero. The output is

$$Z = \tan^{-1} Y/X$$

$$X = K(X^2 + Y^2)^{1/2}$$

$$Y = 0.$$

There are two tests made prior to the first angular iteration. The first test is to compare the input values of Y and X. If the absolute value of Y is greater than the absolute value of X, then the x,y coordinates are rotated ninety degrees in a negative direction. This procedure will keep the truncation error of Y/X to a minimum. Following the test X is examined to insure that it is positive. If it is not positive, the x,y coordinates are

rotated 180 degrees to make X positive.

After the final iterations are done, the value of X is multiplied by $1/K$ in order that X will finally contain $(X^2 + Y^2)^{1/2}$, where K is the elongation factor previously described.

C. OTHER IMPLEMENTATIONS OF CORDIC

Two other approaches to the CORDIC technique are briefly discussed. They are a decimal approach proposed in Ref. 10 and an approach described in Ref. 11 commonly used for hand electronic calculators including the Hewlett-Packard HP-35.

1. Decimal Approach

The CORDIC technique has primarily been implemented in binary form. Schmid and Bogacki [ref. 10] demonstrate a method of using the CORDIC technique in decimal form. For all angular steps after the first, they suggest using the constant number of substeps, nine. This simplifies the programming of the algorithm, but as a result it uses a total of thirty-seven steps to compute trigonometric functions to five-digit accuracy. Using a constant number of substeps is in contrast to the decimal implementation described above. there the number of substeps was minimized, constrained by the convergence criterion and the desired $n-1$ place accuracy.

2. HP-35 Approach

In many small calculators compactness of code is more important than speed. In the HP-35 Calculator the CORDIC algorithm is used to simultaneously generate $K \sin Z$ and $K \cos Z$ where K is the elongation factor. $\tan Z$ is derived from these by division and then $\sin Z$ and $\cos Z$ are derived using identities involving square root and $\tan Z$.

The square root function uses the conventional sum of odd digits algorithm. The advantage of this approach is the lack of any need to use predetermined elongation constants, which is important in a decimal machine where the number of subiterations and the sequence of rotation constants is a function of the data [Ref. 11].

IV. IMPLEMENTATION

Software implementation is divided into two major areas, the maneuvering board solution algorithm and the control program. The maneuvering board solution algorithm is the heart of the software system. To be executed successfully, the maneuvering board algorithm must have inputs which are complete and within certain bounds. An extensive control program is required to protect the user from misuse and to insure all necessary input values are supplied before the maneuvering board solution algorithm can be called. Additionally, the control program facilitates storage of observed data for multiple contacts.

A. HARDWARE CONSIDERATIONS

The concept of solving Navy problems with microcomputers is in no way dependent upon the capabilities and characteristics of a particular manufacturer's product. Several factors, however, led to the selection of the Intel MCS-4 Micro Computer Set as the prototype hardware for the maneuvering board computer. The computer set is relatively inexpensive and was available at the Naval Postgraduate School, along with a good library of associated utility software and an interactive assembler/simulator running under CP/CMS on the School's IBM 360/67, as described in Ref. 12. The Intel product operates using programmable read-only-memory (PROM), allowing construction of a functioning prototype without the added expense of ROM etching. After selection of the MCS-4, several implementation decisions made during development of the system were directly dependent upon hardware features of the set.

The COREIC algorithm is essential to the implementation of the maneuvering board solution described below. COREIC, in turn, requires an efficient shifting operation. The MCS-4 is a hexadecimal machine: binary shifting is cumbersome and time consuming. For this reason, and because input by the user is in the form of decimal digits, numbers are stored and manipulated in decimal character form through use of the machine's decimal adjust instruction. The random-access-memory (RAM) of the MCS-4 is organized and addressable in sixteen-byte blocks or registers. For this reason, and because all values involved in computation fall within a narrow enough range, fixed point representation of numbers was chosen as the simplest and least time consuming for manipulation. The largest input value in any maneuvering board problem is a contact's range, practically limited by search radar to approximately one hundred thousand yards. All numbers, therefore, are represented as sixteen-digit decimals in tens-complement form, seven digits before the decimal point and nine after.

Reference 13 is a complete description of the MCS-4 hardware and a full explanation of its instruction set.

B. MANEUVERING BOARD SOLUTION ALGORITHM

The maneuvering board algorithm was designed to solve the basic relative movement problem described above. With this algorithm closest point of approach problems, stationing problems given time to station, and intercept problems can be solved. The algorithm utilizes conversion between polar and rectangular co-ordinates, together with vector addition and subtraction to solve the basic maneuvering board problem. The decision to use this method rather than using the Law of Cosines and the Law of Sines was based upon the realization that the COREIC algorithm neatly provides the cosine and sine of an input angle in the

rotational mode and $(x^2 + y^2)^{1/2}$ and $\tan^{-1} y/x$ in the vector mode. These functions are necessary in conversion from polar to rectangular co-ordinates and back again. The flow chart for the maneuvering board solution algorithm is found in Appendix F.

1. Solution of the Relative Plot

Let the following be inputs: own ship's course, CO, speed, SO, and two positions in time; that is, bearing, B1, B2, range, R1, R2, and time, T1, T2. The first step is to convert the positions to rectangular co-ordinates:

$$X1 = R1 \cos B1$$

$$Y1 = R1 \sin B1$$

$$X2 = R2 \cos B2$$

$$Y2 = R2 \sin B2.$$

Considering X1, Y1 and X2, Y2 as the x,y components of two vectors, V and W respectively, the relative motion line can be considered as the vector derived from the subtraction of vector V from vector W. Thus the x,y components of the relative motion line are

$$X0 = X2 - X1$$

$$Y0 = Y2 - Y1.$$

Since the quotient $Y0/X0$ represents the slope of the relative motion line, $Y0$ and $X0$ are compared to avoid a possibly infinite slope. If $|Y0|$ is greater than $|X0|$, then the x,y components are rotated ninety degrees and the slope $M = Y0/X0$ is computed. The direction of relative motion (DRM) and the speed of relative motion (SRM) are then found by converting the x,y components of the relative motion line back to polar co-ordinates.

$$DRM = \tan^{-1} (Y0/X0)$$

$$SRM = \frac{(XC^2 + YC^2)^{1/2}}{T2 - T1}$$

The closest point of approach can be solved for in rectangular co-ordinates as the intersection of the relative motion line having slope M and passing through points (X1,Y1) and (X2,Y2) and a line perpendicular to the relative motion line which passes through the origin and has slope -1/M. Specifically,

$$XCPA = \frac{X1 M^2 - Y1 M}{M^2 + 1}$$

$$YCPA = \frac{Y1 - X1 M}{M^2 + 1}$$

The rectangular co-ordinates of the closest point of approach are then converted to polar co-ordinates, yielding the range and bearing of CPA

$$CPA \text{ Range} = (XCPA^2 + YCPA^2)^{1/2}$$

$$CPA \text{ Bearing} = \tan^{-1} (YCPA/XCPA),$$

where the time of CPA is

$$\left[\frac{(XCPA - X1)^2 + (YCPA - Y1)^2}{SRM^2} \right]^{1/2} + T1.$$

2. Solution of the Speed Triangle

Considering that the speed triangle is a triangle of three vectors and that the maneuvering ship's course and speed represent one of these vectors, then the maneuvering ship's course and speed can be found as the resultant of the other two.

The basic ideas described above are used again except that speed replaces distance as the magnitude of the vectors. The polar forms of the two known vectors are transformed to rectangular co-ordinates.

Given own ship's course (CO), own ship's speed (SO), speed of relative motion (SRM), and direction of relative motion (DRM), then

$$X1 = SO \cos CO$$

$$Y1 = SO \sin CO$$

$$X2 = SRM \cos DRM$$

$$Y2 = SRM \sin DRM.$$

Thus the x,y components of the maneuvering ship's vector are $X1 + X2$ and $Y1 + Y2$ where the maneuvering ship's speed is

$$\left[(X1 + X2)^2 + (Y1 + Y2)^2 \right]^{1/2}$$

and the course is

$$\tan^{-1} (Y1 + Y2 / X1 + X2).$$

C. THE CONTROL PROGRAM

The CFA algorithm and other routines invoked as required by the actions of the user are controlled by and imbedded

within a constantly executing, continuously looping control program. The flow of control is determined by the user's depressing one of twenty-five keys. The control loop constantly queries the keyboard input until a key depression is detected, and then branches program control to a corresponding section of code. Upon completion of the selected function, control returns to the keyboard query loop.

Although only five bits are required to specify which key has been depressed, the control program is written for a keyboard with a six bit output, the sixth bit being on when and only when a key is actually depressed. The control program therefore reads two four-bit ROM input ports in sequence, the first containing the indicator bit and one identification bit, the second containing the four remaining identification bits. The five identification bits are used as an index into a table of branch instructions to routines corresponding to the keys.

The control program performs two additional functions. The query loop, while waiting for a key to be depressed, repeatedly invokes a subroutine to display the current digital output. After a key has been depressed, the control loop zeroes the information on a particular ROM output port. One bit on this port is used to indicate to the user an incorrectly selected key. The key routines are heavily interactive, and any key depressed out of proper sequence will cause this bit to be turned on, activating an error light.

Details of the control program's logic are provided in Appendix B.

D. THE DISPLAY INTERFACE

Output to the user is provided in two forms: the error light previously described and a sixteen digit light emitting diode (LED) display. The routine controlling this

display is dependent upon the hardware used. The interface to this routine, however, is determined independently. Prior to invoking this routine, the information to be displayed must be placed in a RAM register in the following manner: the rightmost digit of the display must be the low order digit (digit zero) in the RAM register, and correspondingly the leftmost digit must be in the high order location (digit fifteen). The decimal points of the LED's are lighted by the display routine according to the contents of the sixteen bits of the RAM register's four status characters. The leftmost bit of status character zero being on indicates that the decimal point in front of the low order digit is to be lighted. Correspondingly, the rightmost bit of status character three controls the decimal point in front of the high order character. Finally, the address of the RAM register to be displayed must be placed in the particular CPU index register pair required by the display routine. Since the display hardware is assumed to have no memory, the display routine must be invoked over and over at a rapid rate to achieve sufficient brightness and to avoid flicker.

The particular display routine for the prototype machine is available in the program listing, annotated with an explanation of the basic hardware and how it is used. Another display scheme is described in Ref. 14, included in which is a brief explanation of the problems involved in MCS-4 input/output.

V. CONCLUSION

In designing the software for the MCS-4 prototype, the three most important considerations were execution time, memory size, and user interaction. As a result, the maneuvering board algorithm was written to solve the basic relative motion problem only. Since all of the maneuvering board problems use this basic solution, or at least part of it, the algorithm provides the solution foundation for all related problems.

A. RESULTS

Presently the MCS-4 prototype will handle all input information for ten simultaneous contacts, and it can be used to solve the CPA problem, the stationing problem given time to station, and the intercept problem. These problems are solved in 4.5 seconds of execution for all five items of output information. This relatively fast solution time is largely due to the speed of the transcendental CORDIC functions developed for this computer as described above: the rotation mode yields sine and cosine in approximately 360 milliseconds, while the arctan/modulus function takes about 410 milliseconds. Additionally, output information is displayed, not one item at a time, but in sets of associated information: the course and speed comprise one set, and the CPA range, bearing, and time, the other.

As a demonstration of the capability of this prototype, several example problems were worked on the Maneuvering Board (H.C. 2665-10) and compared with the prototype's output. Differences between the two methods of solution are less than one degree in course or bearing, one minute in time, one hundred yards in range, and one-half knot in

speed. The accuracy of the microcomputer's outputs has also been verified through comparison with the answers provided by a FORTRAN simulation of the maneuvering board algorithm written for debugging purposes during implementation of the prototype. Table II illustrates comparisons for three example CPA problems.

Three additional points require explanation. For CPA solutions where both maneuvering ship positions are exactly the same but at different times, the MCS-4 prototype solution appears to be in error as to the closest point of approach. In this special situation the solution is simply that the maneuvering and reference ship are on the same course and speed and that the maneuvering ship is constantly at CPA. The prototype's software was designed to compute the CPA point as the intersection of two straight line, and therefore will not handle a case where one of the lines is actually a point. The maneuvering board algorithm could be modified to handle this situation, but since this rarely occurs, and since the solution is immediately apparent to the trained operator, it was not incorporated.

The second point pertains to the inability to solve a stationing problem given speed to station. This problem requires solution for the x,y components of the maneuvering ship's vector where the magnitude (speed) of the maneuvering ship's vector is known, but the magnitude of the relative motion vector cannot be obtained using the methods described above. The maneuvering board algorithm finds the speed of relative motion by using the time between observations as a divisor. Since the time on station is not an input value, this time difference is not available. The comparable manual solution technique requires swinging an arc of radius equal to the stationing speed to find its intersection with the relative motion vector, thereby defining the rm vector.

Finally, the ability to solve true wind problems could have been incorporated in the prototype, but the need for additional control logic precluded it in the present

TABLE II

CPA PROBLEMS

	PROBLEM ONE	PROBLEM TWO	PROBLEM THREE
<u>REFERENCE SHIP</u>			
Course	350°	000°	025°
Speed	15 kts	14.5 kts	20 kts
<u>MANEUVERING SHIP</u>			
Position 1			
Time	0910	1400	1400
Bearing	012°	103°	340°
Range	27,600 yds	14,000 yds	27,000 yds
Position 2			
Time	0920	1429	1406
Bearing	000°	148°	330°
Range	22,000 yds	14,000 yds	18,800 yds
<u>MANEUVERING ECARD</u>			
<u>SOLUTION</u>			
CPA Bearing	319°	(past	271°
CPA Range	16,500 yds		9,800 yds
CPA Time	0939	CPA)	1416.8
Course	269°	311°	165°
Speed	20 kts	8.5 kts	28.5 kts
<u>MCS-4 PROTOTYPE</u>			
<u>SOLUTION</u>			
CPA Bearing	318.9°	(past	271°
CPA Range	16,594 yds		9,695 yds
CPA Time	0938.9	CPA)	1416.8
Course	269.3°	310.3°	164.4°
Speed	19.8 kts	8.4 kts	28.3 kts

implementation. True wind problems require no relative plot, and hence would utilize only that portion of the maneuvering board algorithm dealing with the speed triangle.

Examples of stationing given time to station and intercept problems are provided in Appendix A, the users' guide to the prototype machine. Cost estimates are presented in Appendix C.

B. FUTURE WORK

The maneuvering board computer can be a valuable tool to Navy users as implemented. Several extensions are possible, however, in both software and hardware.

Certain software features have not been implemented for lack of time. The ability to display data previously entered could be added most quickly. The ability to compute the reciprocal of a bearing may be of use while entering data in a stationing problem, where own ship's bearing from another ship is required. A more significant improvement would be the ability to compute stationing course given stationing speed instead of time to station, but this requires a second solution algorithm with different inputs as described previously.

The system can be modified to handle more than ten contacts. This would require reallocating one of the sixteen display digits to the contact identification number. Each additional contact requires two additional RAM registers, or half of a RAM chip.

Alternate hardware methods of input/output could significantly improve the maneuvering board microcomputer as a total system. The present keyboard input could be largely supplanted by microcomputer-controlled analog-to-digital converted outputs taken directly from the radar's cursor circuits and a real-time clock. Similarly, output could take the form of a video display whose input is prepared by a microcomputer-controlled digital-to-analog conversion of

the presently available digital output. These features could be implemented inexpensively using independent microprocessors; the maneuvering board computer, handling a more complex problem, requires only thirteen ROM chips, nine RAM's, and a single-chip CPU.

The results of this thesis indicate that microcomputer technology can feasibly be applied to the ideas above and to similar Navy problems.

APPENDIX A

USERS' GUIDE TO MANEUVERING BOARD COMPUTER

This guide to the MCS-4 prototype maneuvering board computer is divided into two sections. The first part consists of instructions to the user and a description of the interactive responses the user may expect in solving CPA problems. The second section contains examples of other types of maneuvering board problems and directions on how the prototype computer can be used for their solution.

A. THE BASIC CPA PROBLEM

In all maneuvering board problems the identification of both a reference ship and a maneuvering ship is required. In a CPA problem the reference ship is the user's own ship, and the maneuvering ship is a contact whose range and bearing are known.

When the machine is first energized, the reinitialization button located on the body of the MCS-4 should be depressed. Then, or at any other time desired by the user, this button erases all stored information, blanks the display, and restarts the internal program.

In the instructions below, keys are identified by the symbols used in Figure 2, the keyboard/display organization schematic. The locations of the various information items in the LED display are indicated in Figure 2 by these same key identification symbols.

There are two types of data which must be entered and stored prior to any solution. The reference ship's data consists of a course and speed. The maneuvering ship's data consists of an identification number, time, bearing, and range for each of two observations.

Certain instructions are common for all six items of input information. Depressing the item's input key terminates the entry of any previous item and enables the number keys; there is no visible interaction. For each item at least one number must be entered before a different input key can be depressed. An incorrectly entered item can be erased from the display by depressing that item's input key again. For each item there is a maximum number of digits which can be entered. All but range and contact identification number can have an additional digit after a decimal point. After at least one digit has been entered, the decimal point key may be depressed, followed by one more digit. Although the user has displayed a decimal point, a digit does not have to follow; a zero will be assumed.

To enter own ship's information, first depress either the speed (S) or course (C) key: these items may be entered in either order. Speed may have up to two digits to the left of a decimal point, course three. Speed is measured in

PROTOTYPE KEYBOARD/DISPLAY SCHEMATIC

LED DISPLAY

						C	C	C	.C				S	S	.S
N	T	T	T	T	.T	B	B	B	.B	R	R	R	R	R	F

KEYBOARD

	CLEAR KEY	
7	8	9
4	5	6
1	2	3
0		.

CPA	TS	
IST	SAVE	N
2ND		T
OS	C	E
COMPUTE	S	R

FIGURE 2

kncts, course in degrees between zero and 360. Leading zeroes are assumed. When course and speed have not yet been entered after a reinitialization, both must be entered at the same time. When course and speed appear correctly on the display, depress the own ship (OS) key. The display will then be blanked, indicating proper storage. Note that once either the C or S key has been depressed, the N, T, B, and R keys cannot be depressed without error until the OS key has been depressed successfully or until the CLEARKEY key is depressed. CLEARKEY clears the display whenever it is depressed, thus erasing any entries which have not been stored. At any time a key is depressed out of acceptable sequence, the error indicator will light and remain lit until a valid key selection is made.

Contact number (N), time (T), bearing (B), and range (R) keys function similarly to C and S. N will take only one digit, between zero and nine. T takes four digits to the left of a decimal point, measuring time on a twenty-four hour clock. The digit after the decimal point represents tenths of minutes and is optional. B takes three digits, again measured in degrees from zero to 360. R takes up to six digits, and is measured in yards. Leading zeroes are assumed. These four items may be entered in any order, and once all four are correctly displayed, they may be stored by depressing either the first (1ST) or second (2ND) observation key, as appropriate. Note that depressing 2ND will generate an error and will not blank the display if a first observation has not already been stored for the contact whose number appears in the display. Again, once one of these four keys is depressed, C and S cannot be depressed without error until 1ST, 2ND, or CLEARKEY has been successfully selected.

To compute CPA information for a contact, first enter only the contact's identification number in the display, then depress the COMPUTE key. If two observations for this

contact and the reference ship's data have been stored previously, the display will be blanked, and after 4.5 seconds the contact's number will reappear along with its time, bearing, and range of CPA. If the time of the contact's CPA is earlier than the time of its second observation, the contact is past CPA and the display will remain blank except for two decimal points. In either case depressing the "targetship" (TS) key will display the contact's course and speed. The CPA key will return the CPA output data to the display. Since CLEARKEY merely erases the display, TS and CPA can be successfully selected after CLEARKEY.

After computation on a contact, either the first or second observation data can be saved to be used as the first observation for a later CPA calculation. Depress either 1ST or 2ND, as appropriate, followed by SAVE. 1ST or 2ND after a COMPUTE, and only then, will blank all of the display except the contact number and display a one or a two as the rightmost digit. SAVE will blank the display and store the designated data as a new first observation. TS and CPA remain enabled. After a COMPUTE, depressing C, S, N, T, B, or R will clear the display and disable CPA, TS, 1ST, 2ND, and SAVE.

Any stored contact observation may be overwritten by the user in the manner of a normal entry described above. After own ship's course and speed have once been entered following a reinitialization, then only one or the other need be entered if either must be changed. In this circumstance, however, all contact observation data is erased. To obtain valid solutions, all observations and computations must be made for a reference ship on a constant course and speed. As a general rule, if observation data and new reference ship data are available for entry at the same time, enter reference ship's data first.

The computer will store data for up to ten contacts. Solutions for the ten contacts may be computed in any order.

E. EXAMPLES OF USE OF THE MCS-4 PROTOTYPE

As indicated above the MCS-4 prototype is primarily used for CPA solutions, but it can also be used for intercept, stationing given time to station, and collision avoidance problems. The sample problems below are used to illustrate how the MCS-4 prototype can be employed.

1. Intercept Problem

Consider a ship contact which is on a course of 200° at a speed of 10 knots. Its bearing and range at time 1010 are 340° and 15,000 yards from own ship. At time 1010 maneuver own ship to intercept the ship contact at time 1100.

The solution procedure is to consider the ship contact as the reference ship and own ship as the maneuvering ship. Enter course 200° , speed 10 knots using the C, S, and OS command keys. Since the contact is the reference ship, enter the reciprocal of the first bearing, 160° , the range of 15,000 yards, the time of 1010, and the contact number. Command keys B, R, T, N, and IST are used. Next enter as the second position a bearing of 160° , a range of one yard, the time of 1100, and the same contact number. Command keys B, R, T, N, and 2ND are used.

To compute the solution, enter the contact number and depress the COMPUTE key. A CPA of zero yards at time 1100 will be displayed. For the intercept course and speed depress the IS command key. The course and speed of 261.7° and 6.5 knots will be displayed.

A solution of 262° and 6.6 knots was obtained when working the above problem on the Maneuvering Board.

2. Stationing Given Time to Station Problem

Consider a ship contact which is on a course of 160°

at a speed of 18 knots. Its bearing and range from own ship are 020° and 16,000 yards at time 1320. At 1320 maneuver own ship to take station so that the ship contact bears 070° at a range of 2,000 yards at time 1350.

Again the solution procedure is to consider the ship contact as the reference ship and own ship as the maneuvering ship. Enter a course of 160° and a speed of 18 knots. Since the ship contact is now the reference ship enter the reciprocal bearing 200° , range 16,000 yards, time 1320, and the contact number for the first position. Next enter the reciprocal bearing 250° , range 2,000 yards, time 1350, and the same contact number for the second position.

For solution results, enter the contact number and depress the COMPUTE command key. Once the CPA results are displayed depress the TS command key to obtain the course and speed to station. The course and speed of 104.7° and 10.0 knots will be displayed.

A solution of 104° and 10.1 knots was obtained when working the above problem on the Maneuvering Board.

3. Collision Avoidance Problems

Collision avoidance problems are situations where a ship contact is determined to be holding a steady bearing or a very slight bearing drift with a decreasing range. In this situation the MCS-4 prototype can be used to determine a course and speed which will open the CPA range. If some specified time is given, for example the time the collision is expected to occur, the collision avoidance problem is exactly a stationing problem given time to station. The ship contact is treated as the reference ship and its reciprocal bearings are entered. The second position data is entered as the desired closest point of approach of the ship contact at the expected collision time.

Note that in solving any of these non-CPA problems, entering a contact's course and speed as the reference

ship's data will, for the reasons described above, erase all the previously entered observation data for other contacts.

APPENDIX B

CONTROL PROGRAM LOGIC DESCRIPTION

the control program is described below in terms of its logical flags and counters and the functions performed when each key is depressed by a user.

A. FLAGS AND COUNTERS

Before a CPA solution can be determined for a particular contact, eight separate inputs must be stored in the computer. The flags and counters described below are used to guide the input of information and the execution of the CPA algorithm.

1. Format Flags

Inputs to the machine are made in one of two formats. The first corresponds to the four pieces of information required to represent the observation of a contact: the contact's identification number, the time of observation, its bearing, and its range. The second format is used for entering the reference ship's course and speed. There is a flag corresponding to each format, set when information making up that format is being entered.

2. Selection Flags

When information is being input to the computer, one of six selection flags will be set, corresponding to the particular item being entered.

3. Input_Flags

Six input flags correspond to the six selection flags. These flags are set when a sufficient number of digits for the current item has been entered.

4. Storage_Flags

A flag is set when the contents of a complete format is entered into storage. One storage flag indicates that the reference ship's course and speed have been entered. The twenty other storage flags, two for each of ten possible contacts, are used to indicate the storage of first and second observations' data.

5. The_Decimal_Point_Flag

A decimal point may be requested only once for a given input item, and a flag is set when this request has been made.

6. The_Solution_Flag

When the CPA algorithm is executed, a solution flag is set indicating that output information is available for display.

7. Counters

Two CPU registers are used as counters indicating the maximum number of digits allowed and the minimum number required for a particular input.

B. THE KEYBOARD

Twenty-five functional and numeric keys are required for control of and input to the computer.

1. Input Keys

There are six input keys corresponding to the six types of input information: contact identification number, time of observation, its bearing and range, and the reference ship's course and speed. Each key performs the following functions:

a. Each verifies that the format flag for the format not corresponding to the key is not set. Otherwise, it indicates an error and returns to the control loop.

b. Each then verifies that the minimum number of digits for the preceding input item has been supplied, or ascertains that the preceding item is also the item just selected. Otherwise, it indicates an error and returns to the control loop.

c. The key then resets all selection flags, the decimal flag, and the input flag corresponding to the item selected. It sets the format flag and the selection flag corresponding to the item selected. If the solution flag is set, it resets that flag and blanks the entire display; otherwise, it blanks that portion of the display corresponding to the item selected.

d. Finally, it initializes various CPU registers as required to prepare the display for the digits which will follow.

2. Storage Keys

The three storage keys correspond to the first and second observations of a contact, and to the reference ship's course and speed data. The following common functions are performed by each of the three keys:

a. Each verifies that all input flags required by the corresponding format are set. Otherwise, it indicates an error and returns to the control loop.

b. It stores the information, setting the appropriate storage flag.

c. It blanks the display.

There are significant differences among the storage key routines. As a protection against error by the user, information for a second observation will not be stored unless data from a first observation has already been entered. Otherwise, an error condition will be signaled.

The reference ship storage function is complicated by two factors. First, once the reference ship's course and speed have been entered, the user may desire to change only one or the other. Consequently, if the course and speed storage flag is already set, only one or the other item is required. Second, the CPA algorithm is based on all observations being made while the reference ship maintains a constant course and speed. Consequently, if the reference ship course and speed storage flag is set, entering a course or speed change will reset the storage flags for all observations on all contacts.

Finally, the first and second observation keys have a secondary function in connection with the save key described below. Consequently, if the solution flag is set, depressing one of these keys will load a CPU register with a value indicating which key has been depressed, and the normal function of the key will be suppressed.

3. The Compute Key

The following functions are performed by the compute key:

a. It verifies that only a contact identification number is presently entered in the display, and that storage flags for the reference ship's data and both observations for the indicated contact are set. Otherwise, it indicates an error and returns to the control loop.

b. It transfers contact observation data and reference ship's data to RAM register locations as required by the CPA algorithm, and executes the algorithm.

c. It sets the solution and reference ship's course and speed flags, resets all other flags except observation flags for other contacts, and restores the reference ship's data to the RAM register location required by the control program.

d. It zeroes all counters, and reinitializes the CPU registers as required by the control program.

e. Finally, it displays the contact's number and the time, range, and bearing of its CPA.

4. Output Keys

The two output keys first verify that the solution flag is set, and then display the requested information. Otherwise an error will be indicated. The CPA key displays the contact identification number and the time, range, and bearing of its CPA. The "targetship" key will display the contact's identification number and its course and speed.

5. The Save Key

After executing the CPA algorithm for a selected contact, the user has the option of saving the contact's first or second observation data for future use. The save key performs the following functions:

a. It verifies that the solution flag is set and that either the first or second observation key has been depressed. Otherwise, it indicates an error and returns to the control loop.

b. If the second observation key has been depressed, it transfers the second observation data to the solution contact's first observation RAM location, thus destroying the first observation information.

c. It resets the first observation location storage flag for the solution contact.

6. Clear Display Key and Reinitialization Button

The clear display key blanks the display and resets all flags except storage flags. This is to be distinguished from the reinitialization button, a hardware feature of the MCS-4 microcomputer set which first zeroes memory and all CPU registers, and then restarts the program.

7. The Decimal Point Key

The decimal point key verifies that the decimal point flag is not set and that the minimum number of digits required for the current input item have been entered. Otherwise, an error is indicated and the control returns to the control loop. The decimal point is then displayed in a position corresponding to the present input, and the decimal point flag is set.

8. The Number Keys

There are ten digit keys, each performing the same functions:

a. Each verifies that digits may be entered by inspecting the minimum digit counter: unless a selection key has been depressed, this counter will indicate that the maximum number of digits has already been entered. Otherwise, it indicates an error and returns.

b. It increments the maximum digit counter.

c. If the decimal point flag is set, it displays the newly selected digit to the right of the decimal point. Otherwise, it enters the new digit to the left of the decimal point, shifting the already entered digits left.

d. If the minimum number of digits has not already been entered, it increments the minimum digit counter. If the minimum number of digits has now been achieved, it sets the input flag for the item being entered.

APPENDIX C

ESTIMATES OF MANUFACTURING COSTS

A. Estimated Cost Breakdown of the MCS-4 prototype

COMPONENT	QUANTITY	\$ COST PER UNIT
Power Supply (+15v,-10v)	1	60
Clock	1	10
1702 FROM	13	45
4002 RAM	9	30
4004 CPU	1	60
4008 Memory Interface	1	30
4009 Memory Interface	1	30
P.C. Board	1	75
Display	1	80
Package	1	75
Total Cost		\$1275

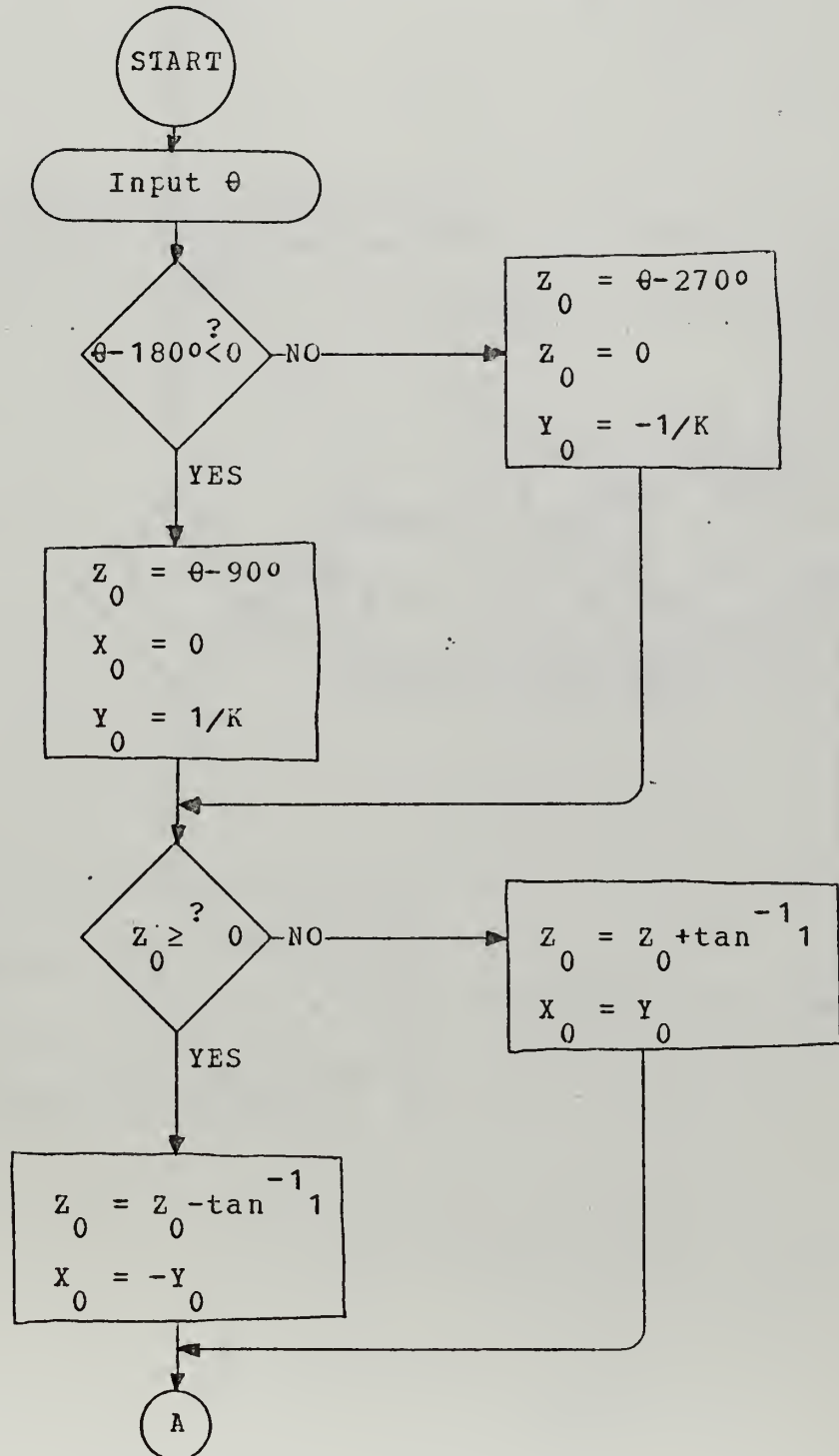
B. Estimated Cost Breakdown of Mass-Produced Computers

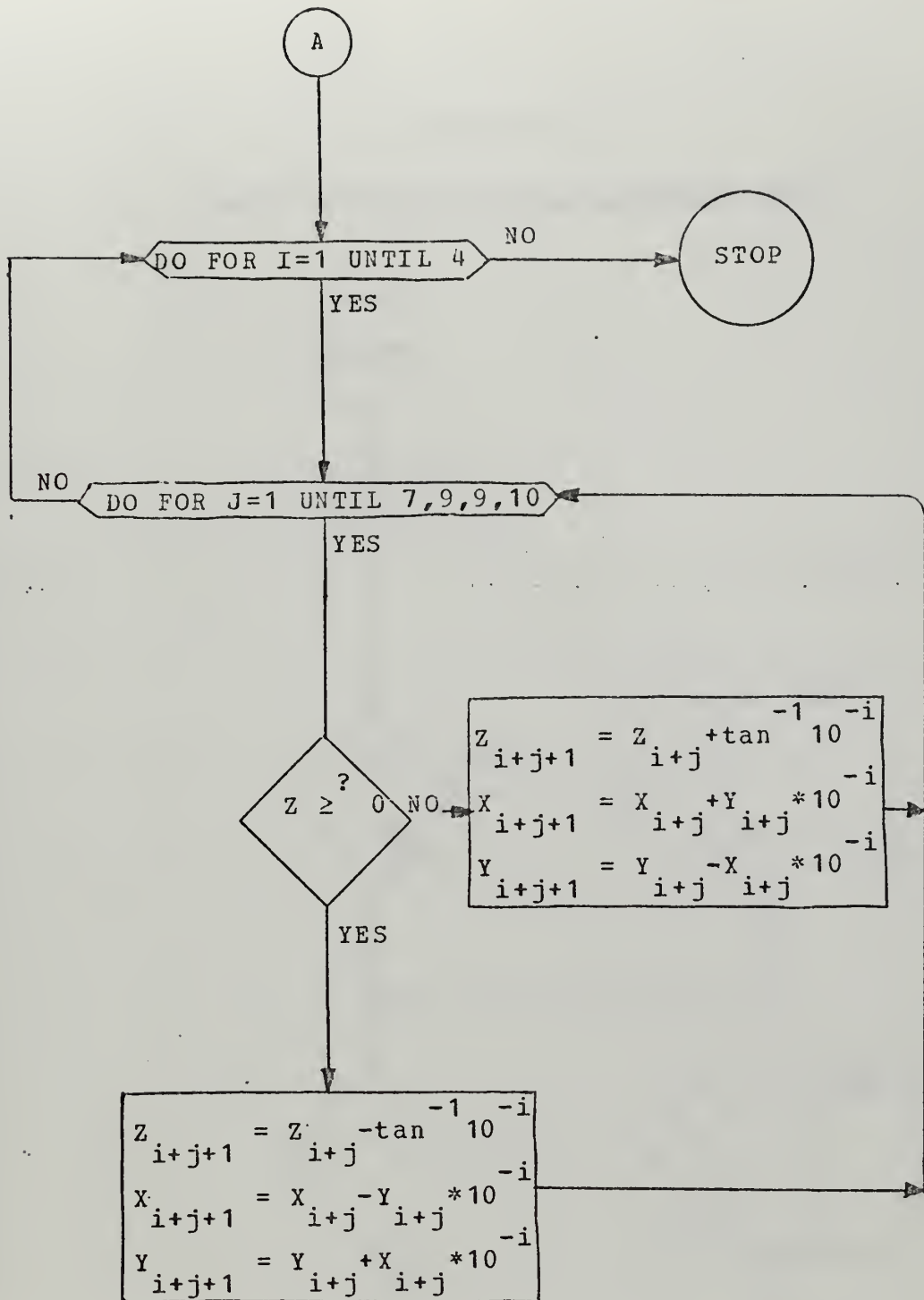
COMPONENT	\$ COST PER UNIT IN QUANTITY RANGE		
	25-99	100-999	1000 PLUS
Power Supply	40	25	15
Clock	10	5	3
4001 ROM (13)	25.50	15	5
4002 RAM (9)	21	15	5
4004 CPU	42	30	20
P.C. Board	25	10	7
Display	50	30	20
Package	40	25	15
Computer Total Cost	\$727.50	\$455.00	\$190.00

Note: A six hundred dollar initial etching cost per ROM, totaling \$7800 regardless of quantity, was not included in the above cost estimates for mass production.

APPENDIX D

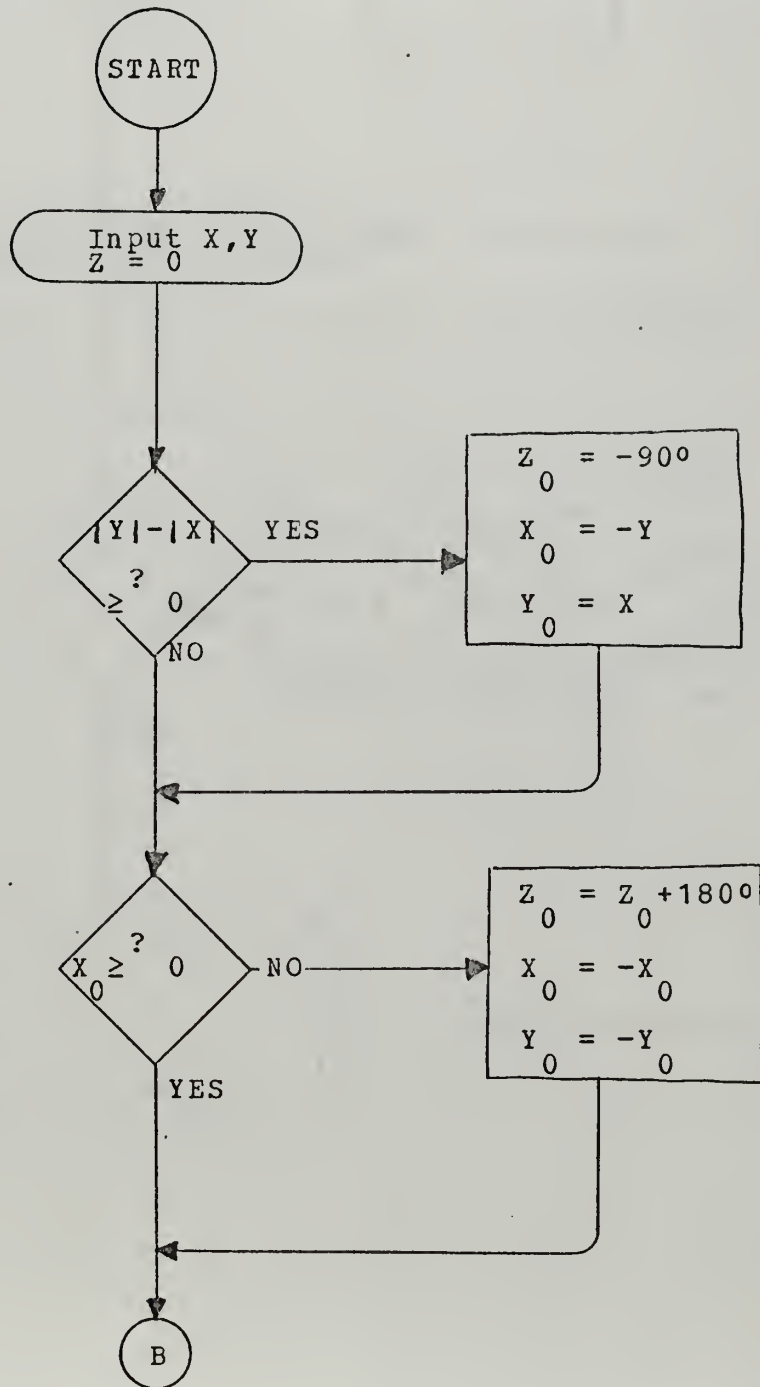
FLOW CHART FOR CORDIC ROTATION MODE

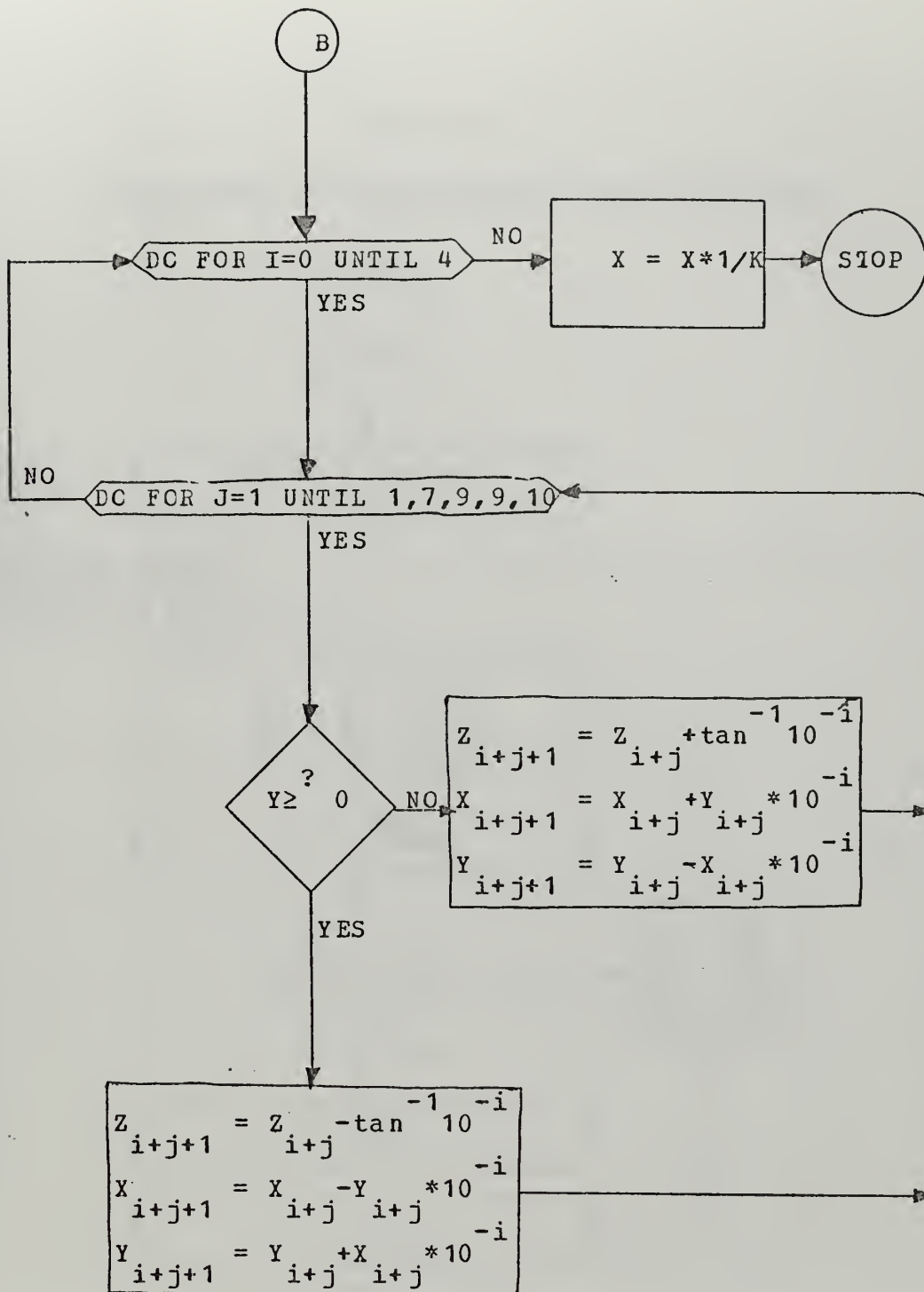




APPENDIX E

FLOW CHART FOR CORDIC VECTORING MODE

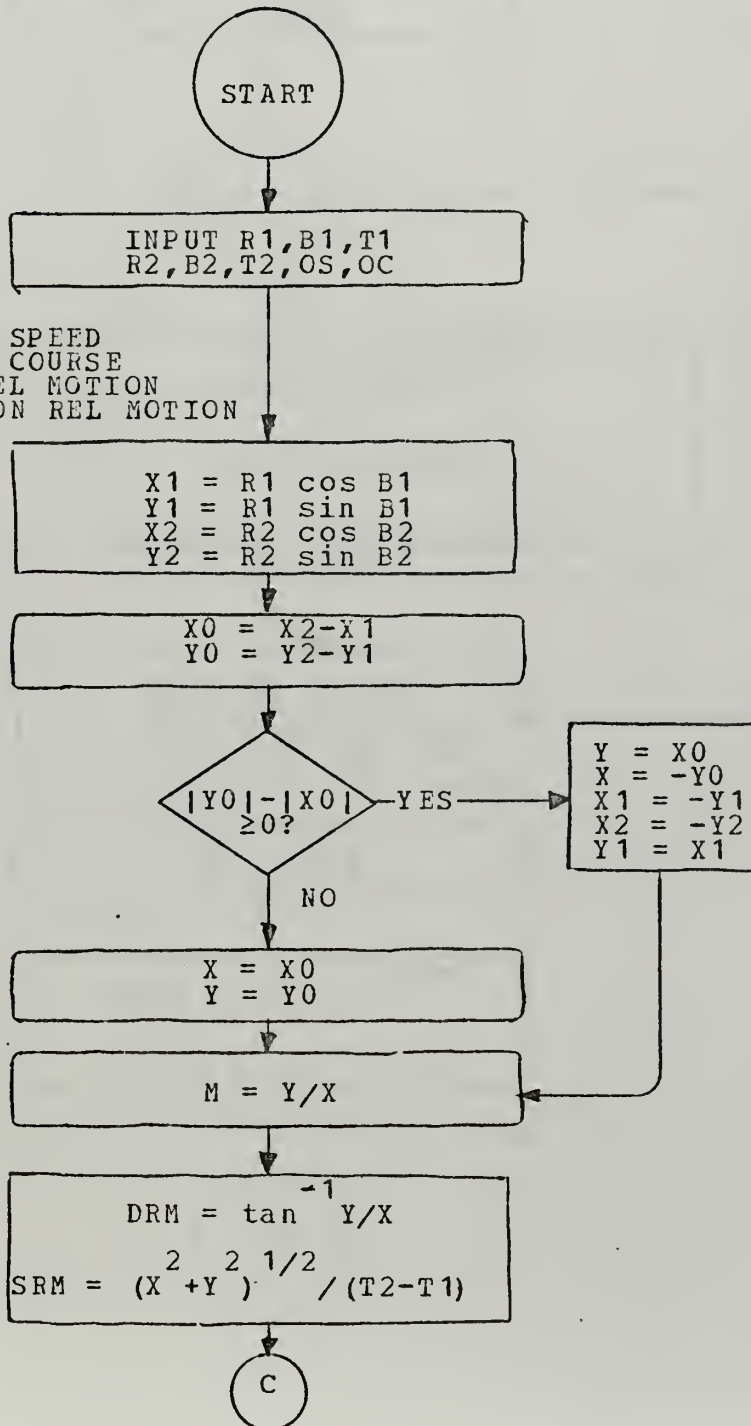


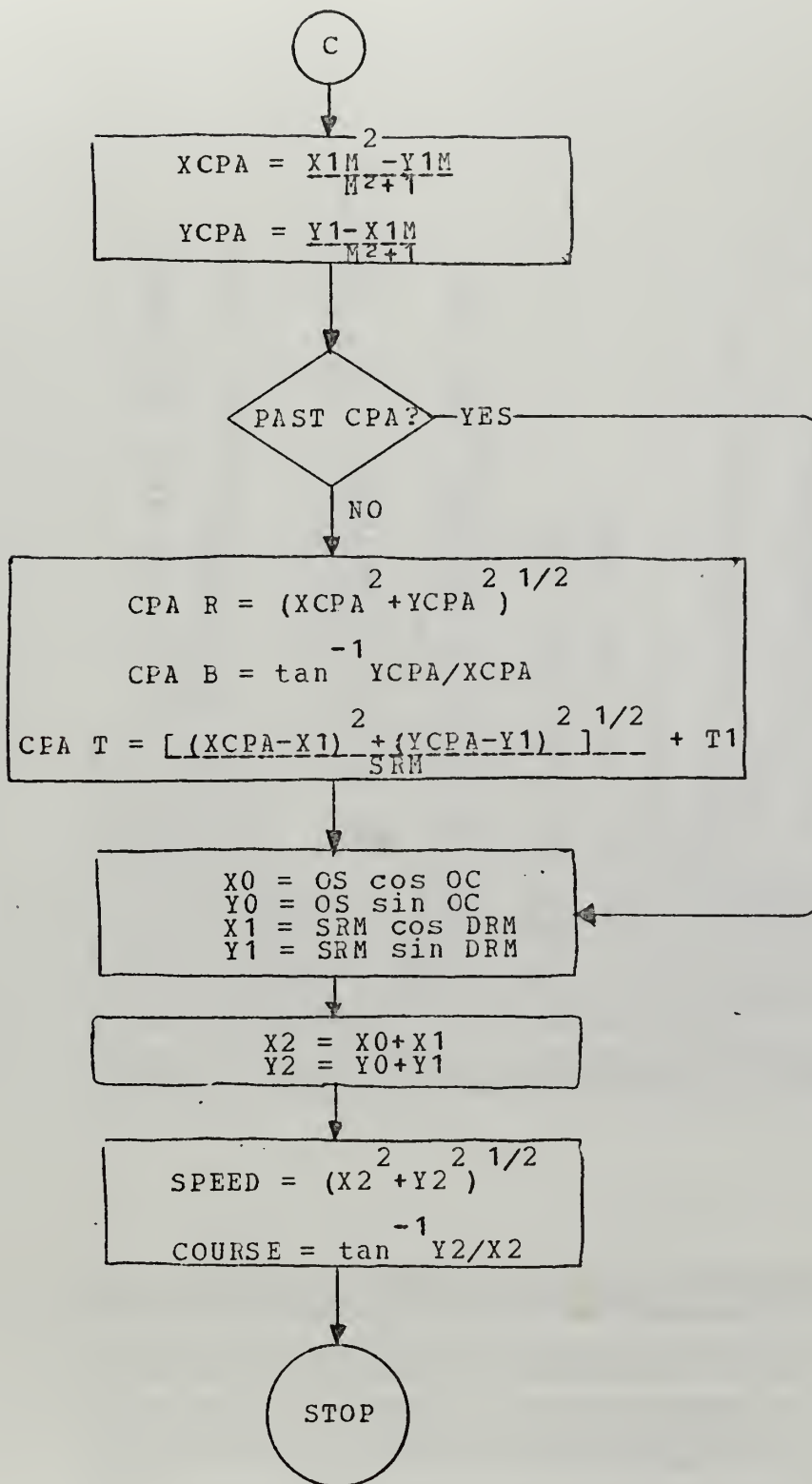


APPENDIX F

FLOW CHART FOR MANEUVERING BOARD ALGORITHM

KEY:
R=RANGE
B=BEARING
T=TIME
M=SLOPE
OS=OWN SHIP SPEED
OC=CWN SHIP COURSE
SRM=SPEED REL MOTION
DRM=DIRECTION REL MOTION





*

```

REM; CONTROL PROGRAM;
NOP;
REM; BLANK DISPLAY;
REM; RC "00"; LDM O; XCH RB; LDM "F"; JMS DCRAM;
REM; TURN OFF ERROR LIGHT;
REM; RC "40"; SRC RC; LCM O; WRR;
REM; FIRST TIME; MAKE SURE KEY NOT ALREADY CCWN;
REM; CHECK NO; CHECKSTROBE;
REM; DISPLAY; KEY TO BE CEPRESSED; A ZCY CHECKSTROBE;
REM; WAIT FOR; SRC RC; RCR; RAL; JCN ZCY CHECKSTROBE;
REM; WAIT FOR; KEY TO STCP BOUNCING;
REM; WAIT 5; KEY'S 4-BIT DATA;
REM; STORE; SRC RA; RCR; XCH RD;
REM; RA "00"; MAKE SURE KEY STILL CCWN; HENCE DATA GOOD;
REM; MAKE SURE KEY STILL CCWN; CHECKSTROBE;
REM; RC; RDR; RAL; JCN ZCY CHECKSTROBE;
REM; SRC RC;
REM; TURN OFF ERROR LIGHT;
REM; RA "40"; SRC RA; LCM O; WRR;
REM; LD RM;
REM; USE DATA BITS AS TABLE INDEX;
REM; RO KEY TAB;
REM; RAL; JCN ZCY RC; NUMBERSIDE;
REM; INC; RO; INC RC; RD;
REM; CLC; ADD RD; JCN ZCY DC1; INC RO;
REM; DC1: CLC; ADD RI; XCH RI; JCN ZCY DC2; INC RO;
REM; DC2: JUMP INTC TABLE;
REM; JIN RO;
REM; ALL TABLE; ROUTINES RETURN HERE; "10"; SRC RC;
REM; CHECK NO; CHECKSTROBE; JMS DISPLAY; FIM RC "10"; SRC RC;
REM; MAKE SURE KEY HAS BEEN RELEASED;
REM; RDR; RAL; JCN LCCP;
REM; IF YES; JCN LCCP;
REM; CHECKSTROBE;
REM; THE TABLE;

```



```

KEYTAB: CKNOSTRCBE;
JUN  CHEMBER;
JUN  NUMBER;
JUN  NUMBER;
JUN  NUMBER;
JUN  NUMBER;
JUN  NUMBER;
JUN  NUMBER;
JUN  NUMBER;
JUN  CHECKKNCSTRCBE;
JUN  DECIMAL;
JUN  CHECKKNCSTRCBE;
JUN  CLEARKEY;
JUN  CLEARKEYSTRCBE;
JUN  CHECKKNCSTRCBE;
JUN  COMPUTER;
JUN  CWSHIP;
JUN  COURSE;
JUN  SAVE;
JUN  SPEED;
JUN  RANGE;
JUN  BEARGETSHIF;
JUN  TARGETKNCSTRCBE;
JUN  CPA;
JUN  KNCSTRCBE;
JUN  CHECKKNCSTRCBE;
JUN  SECOND;
JUN  TIME;
JUN  TARGETNUMBER;

```

```

SYN BLKTO=10, REGTO=12, PREGTO=13, BLKFM=11, REGFM=6,
PREGFM=7, TEMP=9;

REM TO TRANSFER FROM CAE RAM BLOCK TO ANOTHER;
BLCKTRANS;
LD BLKFM; DCL; SRC REGFM; RDM; XCH TEMP;
LD BLKTO; DCL; SRC REGTC; LD TEMP; WRM;
INCR PREGFM; ISZ PREGTC BLCKTRANS; BBL 0;

SYN BLKTO, REGTC, PREGTC, BLKFM, REGFM, PREGFMTEMP;

SYN CONTR=6, TARGNR=11, BLKCODE=10, REGNR=12, PREGNR=13;

REM TO CONVERT TARGET NUMBER TC STORAGE LOCATION;

```



```

BLCKSETUP: SUB TARGNR; JCN NZCY BLK1;
CLC; LDM 7; TARGNR; LDM 2; JCN BLKCCDE; JUN BLKJOIN;
CMMA; XCH TARGNR; XCH BLKCCCE;
BLK1: LDM 1; XCH BLKCCCE; TARGNR; ADD CNTNR;
BLKJOIN: LD TARGNR; ADD TARGNR; ADD CNTNR;
XCH REGNR; LCM 0; XCH FREGNR; BBL 0;
SYN CNTNR, TARGNR, BLKCCCE, REGNR, PREGNR;

REM CHANGES F'S TO O'S;
UNF: RC; RCM; CMA; JCN NZAC FOK; LDM 0; WRM;
SRSC: ISZ RD UNF; BBL 0;
EFCK:

REM DETERMINES IF ENTIRE DISPLAY CR ONLY PART
MUST BE BLANKED;
TEST SOL: SRC RC; RCM; JCN ZAC TSL;
FFIM RC "20"; SRC RC; RCM; JCN ZAC TSL;
LDM 0; WRM;
LDM 0; WRM;
FFIM RC "00"; LCM 0; XCH RB; LDM "F"; JMS DCRAM;
FFIM C; WRO; WRI; WR2; BBL 0;
TS1: FIM RC "CC"; LD RA; XCH RD; LDM "F"; JMS DCRAM;
BBL 1;

REM BLANKS DISPLAY AND INSERTS NUMBER OF
CBSERVATION TO BE SAVED;
SEESAVE:
XCH R6;
XCH RC "00"; LCM 1; XCH RB; LDM "F"; JMS DCRAM;
FFIM RC "00"; LD R6; SRC RC; WRM;
FFIM 0; WRO; WRI; WR2;
JUN CHECKNCSTRCBE;

BEGIN 256;
REM RETURNS TO MAIN RAM BANK AND
TURNS ON ERRCR LIGHT;
BB1:
LDM 0; DCL;
LDM RA "40"; SRC RA; LCM 15; WRR;
JUN CHECKNCSTRCBE;

REM FUNCTIONS PERFORMED BY KEY ROUTINES
DESCRIBED IN APPENDIX B.

FLAGS AS FOLLOWS:
10 - OWN SHIP COURSE AND SPEED STORAGE FLAG
11 - OBSERVATION FCRMT FLAG
12 - COURSE AND SPEED FORMAT FLAG
13 - DECIMAL POINT FLAG

```


01366	01367	01368	01369	01370	01371	01372	01373	01374	01375	01376	01377	01378	01379	01380	01381	01382
02665	02666	02667	02668	02669	02670	02671	02672	02673	02674	02675	02676	02677	02678	02679	02680	02681
01109	01110	01111	01112	01113	01114	01115	01116	01117	01118	01119	01120	01121	01122	01123	01124	01125
109	109	109	109	109	109	109	109	109	109	109	109	109	109	109	109	109

```

CONTACT ID NUMBER SELECTION FLAG
RANGE SELECTION FLAG
TIME SELECTION FLAG
BEARING SELECTION FLAG
COURSE SELECTION FLAG
SPEED SELECTION FLAG
CONTACT ID NUMBER INPUT FLAG
RANGE INPUT FLAG
TIME INPUT FLAG
BEARING INPUT FLAG
COURSE INPUT FLAG
SPEED INPUT FLAG
SOLUTION FLAG;
COMPUTE:
FIM RC "10"; SRC RC; RCM; JCN ZAC BBL;
FIM RC "12"; SRC RC; RCM; JCN NZAC BBL;
FIM RC "14"; SRC RC; RCM; JCN ZAC BBL;
FIM RC "1A"; SRC RC; RCM; JCN ZAC BBL;
LDM 13; XCH RB; RD; SRC RC; RDM; JCN NZAC BBL;
COMCHECK: INC RD;
ISZ RB; CCMCH R6; SRC RC; RCM; XCH RB;
FIM RC "0F"; WRM;
LDM "F"; WRM;
JMS BLOCK; KSETUP;
LD RA; SRC; JCN ZAC BBL;
SRC RC; SRC; RCM; JCN ZAC BBL;
INC RC; WRO;
LDM 0; XCH R6; LD RD; XCH R7; LD RA; XCH RB;
LDM RC; XCH RA; FIM RC "EC"; JMS BLOCKTRANS;
LD R6; DAC; XCH R6; SRC R6; LDM 0; WRO; FIM RC "F0";
JMS BLOCKTRANS; FIM RC "EC"; JMS STATUSIN;
FIM RA "D6"; FIM RC "F0"; JMS STATUSIN;
JUN CPACOMPUTE;
CPA RETURN; FIM RC "DC"; JMS STATUSOUT;
FIM RA "E0"; FIM RC "D6"; JMS STATUSOUT;
FIM RA "F0"; SRC RC; LCM 15; WRM;
INC RD; JMS ZERCRAM;
INC RC; XCH R2; LDM 15; WRM;
LDM 0; XCH R2; FIM R6 "CC"; JMS TRANSLCCP;
FIM R4; WRO; WRM;
FIM 0; WRO;
FIM RE "CO";
JUN CHECKNO$TROBE;

```


0231	0232	0233	0234	0235	0236	0237	0238	0239	0240	0241	0242	0243	0244	0245	0246	0247	0248	0249	0250	0251	0252	0253	0254	0255	0256	0257	0258	0259	0260	0261	0262	0263	0264	0265	0266	0267	0268	0269	0270	0271	0272	0273	0274	0275	0276	0277	0278		
0558	0562	0565	0570	0572	0575	0581	0583	0585	0591	0592	0596	0602	0610	0616	0623	0626	0628	0634	0637	0643	0645	0650	0654	0659	0663	0669	0673	0679	0682	0684	0686	0688	0692	0696	0702	0705	0712	0716	0716	0716	0716	0716	0716	0716	0716	0716	0716		
02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02	02
22E	232	235	23A	23C	23F	247	249	24F	250	254	25A	25F	262	268	26F	272	274	274	27A	27D	283	28A	28E	293	297	29D	2A1	2A1	2A7	2AA	2AC	2AE	2B0	2B4	2B8	2BE	2C5	2CC	2CC	2CC	2CC	2CC	2CC	2CC	2CC	2CC	2CC	2CC	2CC

```

VITIATE1: SRC RC; WRO; ISZ RC VITIATE1;
LDM 2; DCL; LCM 0; WRO; INC RC; ISZ RD VITIATE2;
VITIATE2: SRC RC; WRO; INC RC; ISZ RD VITIATE2;
LDM 0; DCL; ZAC JOINOS;
LD R4; JCN "00"; FIM RC "00"; LDM 13; XCH RF;
FIM RA "00"; FIM RC "00"; LDM 13; XCH RF;
JMS TRANSPART;
JUN JOINCS;
NEEDBOTH: FIM RC "1E"; SRC RC; RCM; JCN ZAC BB2;
INC RD; RCM; JCN ZAC BB2; JMS TRANSLLOOP;
FIM R4; "00"; FIM R6 "DC"; SRC RC; LDM "F"; WRM;
JOINCS: FIM RC "10"; SRC RC; LDM "F"; WRM;
INC RD; JMS ZERORAM;
FIM RC "DO"; JMS UNF; FIM R6 "00";
FIM RC "00"; LCM 0; XCH RB; LDM "F"; JMS DORAM;
LDM 0; WRO; WRI;
JUN CHECKNOSTRCBE;

COURSE: "11"; SRC RC; RCM; JCN NZAC BB2;
FIM RC; JCN ZAC CCURSECK;
LD R6; "18"; SRC RC; RCM; JCN ZAC BB2;
FIM RC; JMS UNSET7; LCM "F"; WRM;
FIM RC "12"; SRC RC; WRM; 0; WRM;
FIM RC "18"; SRC RC; WRM; 0; WRM;
FIM RA "1E"; JMS TESTTSCCL; FIM R4 "0D";
FIM RA "6C"; FIM R6 "FD";
LDM 0; WRI; FIM R8 "07"; JCN CHECKNCCSTROBE;
FIM R8

FIRST: "20"; SRC RC; RCM; JCN ZAC SETCHECK1;
FIM RC; JUN SEESAVE;
LDM 1; JMS SAME12;
SETCHECK1: JMS SAME12;
JMS BLOCKSETUP;
SAMEFS: XCH RB;
LDM 0; JMS BLOCKTRANS;
FIM R6 "00"; LDM 15; WRO;
JMS UNF; DCL;
LDM 0; "11"; JMS ZERORAM; LDM "F"; JMS DORAM;
FIM RC "00"; LCM 0; XCH RB; WRI;
LDM C; "00"; JCN CHECKNCCSTROBE;
FIM R6

SECOND: "20"; SRC RC; RCM; JCN ZAC SETCHECK2;
FIM RC

```



```

FIM RC "17"; SRC RC; RCM; JCN ZAC BB3;
BEARINGOK;
JMS UNSET7; SRC RC; LCM "F"; WRM;
FIM RC "11"; SRC RC; WRM;
FIM RC "17"; SRC RC; WRM;
FIM RC "1D"; SRC RC; LCM O; WRM;
FIM RA "6C"; JMS TESTSCL;
LDM O; WRL;
FIM R6 "FD"; FIM R4 "OD";
FIM R8 "07";
JUN CHECKNOSTRCBE;

TIME: RC "12"; SRC RC; RCM; JCN NZAC BB3;
LD R6; JCN ZAC TIMEOKCK;
FIM R6; SRC RC; RCM; JCN ZAC BB3;
TIMEOKCK: JMS UNSET7; LCM "F"; WRM;
FIM RC "11"; SRC RC; WRM;
FIM RC "16"; SRC RC; WRM;
FIM RC "1C"; SRC RC; LCM O; WRM;
FIM RA "AB"; JMS TESTSCL; FIM R4 "OC";
LDM O; WRL;
FIM R8 "0B"; JCN CHECKNOSTRCBE;

SPEED: "11"; SRC RC; RCM; JCN NZAC BB3;
FIM RC JCN ZAC SPEEDOK;
LD R6; SRC RC; RCM; JCN ZAC BB3;
FIM RC "19"; SRC RC; WRM;
SPEEDOK: JMS UNSET7; LCM "F"; WRM;
FIM RC "12"; SRC RC; WRM;
FIM RC "19"; SRC RC; WRM;
FIM RC "1F"; SRC RC; LCM O; WRM;
FIM RA "OD"; JMS TESTSCL; FIM R4 "OE";
LDM O; WRL;
FIM R8 "01"; JCN CHECKNOSTRCBE;

REM ZEROES FLAGS 13 THRU 19;
UNSET7:
FIM RC "13"; LDM 9; XCF R8; LDM O;
JMS DORAM; BBL O;

BEGIN 1024;
BB4: RA "40"; SRC RA; LCM 15; WRR;
FIM CHECKNOSTRCBE;

SAVE:
FIM RC "20"; SRC RC; RCM; JCN ZAC BB4;

```

```

366 03 0876 03 0328 03 0330 03 0331 03 0332 03 0333 03 0334 03 0335 03 0336 03 0337 03 0338 03 0339 03 0340 03 0341 03 0342 03 0343 03 0344 03 0345 03 0346 03 0347 03 0348 03 0349 03 0350 03 0351 03 0352 03 0353 03 0354 03 0355 03 0356 03 0357 03 0358 03 0359 03 0360 03 0361 03 0362 03 0363 03 0364 03 0365 03 0366 03 0367 03 0368 03 0369 03 0370 03 0371 03 0372 03 0373 03 0374 03
366 03 0876 03 0328 03 0330 03 0331 03 0332 03 0333 03 0334 03 0335 03 0336 03 0337 03 0338 03 0339 03 0340 03 0341 03 0342 03 0343 03 0344 03 0345 03 0346 03 0347 03 0348 03 0349 03 0350 03 0351 03 0352 03 0353 03 0354 03 0355 03 0356 03 0357 03 0358 03 0359 03 0360 03 0361 03 0362 03 0363 03 0364 03 0365 03 0366 03 0367 03 0368 03 0369 03 0370 03 0371 03 0372 03 0373 03 0374 03
63 03 0876 03 0328 03 0330 03 0331 03 0332 03 0333 03 0334 03 0335 03 0336 03 0337 03 0338 03 0339 03 0340 03 0341 03 0342 03 0343 03 0344 03 0345 03 0346 03 0347 03 0348 03 0349 03 0350 03 0351 03 0352 03 0353 03 0354 03 0355 03 0356 03 0357 03 0358 03 0359 03 0360 03 0361 03 0362 03 0363 03 0364 03 0365 03 0366 03 0367 03 0368 03 0369 03 0370 03 0371 03 0372 03 0373 03 0374 03
63 03 0876 03 0328 03 0330 03 0331 03 0332 03 0333 03 0334 03 0335 03 0336 03 0337 03 0338 03 0339 03 0340 03 0341 03 0342 03 0343 03 0344 03 0345 03 0346 03 0347 03 0348 03 0349 03 0350 03 0351 03 0352 03 0353 03 0354 03 0355 03 0356 03 0357 03 0358 03 0359 03 0360 03 0361 03 0362 03 0363 03 0364 03 0365 03 0366 03 0367 03 0368 03 0369 03 0370 03 0371 03 0372 03 0373 03 0374 03

```


0471	LD D2;	XCH D1;	LD D3;	XCH D2;	
0472	LDM L2;	XCH L1;	ISZ L2	CLK;	
0473	SRC CNT1;	LDM I3;	WRR;	LDM 9;	WRR;
0474	BBL 0;				
0475	SYN PI,	T1,D0,D1,D2,D3,BCCL,CNT1,L1,L2,RAM1,RAM2;			
0476					
0477	CLEARKEY:				
0478	FIM RC	"00";	LDM 0;	XCH RB;	LDM "F"; JMS DCRAM;
0479	LDM 0;	WRO;	WRI;	WR2;	
0480	FIM RC	"11";	JMS ZERORAM;		
0481	FIM R6	"00";			
0482	JUN CHECKNCS	TRCBE;			
0483	REM END	CONTRCL	PROGRAM;		
0484					
0485	BEGIN 1280;				
0486	REMANEUVERING	BOARD	ALGORITHM;		
0487					
0488	REM INPUT	TIMES	RANGES	BEARINGS.	CONVERT TC
0489	RECTANGULAR	COORDS;			
0490	CPACOMPUTE:				
0491	FIM RA	"F6";	FIM RC	"00";	JMS LCADBRG;
0492	JMS	CORDIC;			
0493	FIM RA	"F0";	FIM RC	"3C";	JMS LCADRNG;
0494	FIM R2	"30";	FIM R4	"1C";	FIM R6 "80"; JMS MLRT;
0495	FIM R2	"30";	FIM R4	"2C";	FIM R6 "90"; JMS MLRT;
0496	FIM RA	"E6";	SRC RA;	LCN 0;	WR3; FIM RC "00";
0497	JMS	LCADBRG;			
0498	JMS	CORDIC;			
0499	FIM RA	"E0";	FIM RC	"3C";	JMS LCADRNG;
0500	FIM R2	"30";	FIM R4	"1C";	FIM R6 "60"; JMS MLRT;
0501	FIM R2	"30";	FIM R4	"2C";	FIM R6 "70"; JMS MLRT;
0502	FIM R4	"60";	FIM R6	"10";	JMS TRANSLCCCP;
0503	FIM R4	"70";	FIM R6	"2C";	JMS TRANSLCCCP;
0504	REM COMPUTE	X,Y	COORDINATES	OF	RELATIVE
0505	MOTION	LINE;			
0506	FIM R0	"10";	FIM R2	"80";	JMS SBRT;
0507	FIM R0	"20";	FIM R2	"9C";	JMS SBRT;
0508	REM IF	Y	COORD	>	X
0509	TO	AVOID	PCSSIBLE	INFINITE	SLOPE
0510	RELATIVE	MOTION	LINE;		
0511	FIM R4	"20";	FIM R6	"A0";	JMS TRANSLCCCP;
0512	RDO;	RAR;	JCN	ZCY	POSX;
0513	FIM R0	"A0";	JMS	CPLRT;	
0514	POSX;	FIM R4	"10";	FIM R6	"B0"; JMS TRANSLCCCP;
0515	RDO;	RAR;	JCN	ZCY	POSX;
0516	FIM R0	"B0";	JMS	CPLRT;	


```

POSX: FIM R0 "AO"; FIM R2 "BO"; JMS SBRT;          "70";
RARD0; RAR; JCEN ZCY NORCTATE;
JMS REVOLVE;

REM COMPUTE SLCPE OF RELATIVE MOTION LINE;
NCRCTATE: FIM R2 "20"; FIM R6 "10"; FIM R8
FFIM R4 "AO"; JMS DVRT;

REM COMPUTE DRN AND SRM;
CORDICAT; FIM RC "3C"; JMS LCADTIMELO;
FIM RA "EA"; FIM RC "4C"; JMS LCADTIMEELG;
FIM RA "FA"; FIM RO HRSFERMIN; JMS FEERT;
FIM R4 "30"; FIM R4 "5C"; FIM R6 "AQ"; JMS MLRT;
FIM R2 "40"; FIM R4 "5C"; FIM R6 "BO"; JMS MLRT;
FIM R2 "ED"; FIM RA "AC"; JMS LCADTIMEHI;
FIM RA "FD"; FIM RC "BC"; JMS LCADTIMEHI;
FIM R0 "AO"; FIM R2 "BC"; JMS SBRT;
RD0; RAR; JCEN ZCY TIMECK; JMS FERT;
FIM R4 "CU"; FIM R0 DAY; JMS ADRT;
FIM R0 "AO"; FIM R2 "CO"; JMS ADRT;

REM COMPUTE CPA POINT IN RECTANGULAR COORDINATES;
TIMECK: FIM R2 "20"; FIM R6 "AO"; FIM R8 "CO";
FIM R4 "CO"; JMS DVRT;
FIM R4 "70"; FIM R6 "50"; JMS TRANSLCCP;
FIM R2 "70"; FIM R4 "5C"; FIM R6 "DO"; JMS MLRT;
FIM R2 "80"; FIM R4 "DC"; FIM R6 "50"; JMS MLRT;
XCH R5; SRC R4; LCM 1; WRM;
FIM R2 "90"; FIM R4 "7C"; FIM R6 "30"; JMS MLRT;
FIM R0 "50"; FIM R2 "2C"; JMS SBRT;
FIM R2 "80"; FIM R4 "7C"; FIM R6 "40"; JMS MLRT;
FIM R0 "90"; FIM R4 "30"; JMS TRANSLCCP;
FIM R2 "50"; FIM R6 "4C"; JMS FERT;
FIM R4 "40"; JMS DVRT; FIM R8 "10";
FIM R2 "30"; JMS DVRT; FIM R8 "20";
FIM R4 "40"; JMS DVRT; FIM R8 "20";
FIM R4 "60"; FIM R6 "AQ"; JMS TRANSLOOP;
FIM R0 "AO"; FIM R2 "8C"; JMS SBRT;
RD0; XCH RE;

REM TEST FOR FAST-CPA;
FIM R4 "10"; FIM R6 "AQ"; JMS TRANSLCCP;
FIM R2 "60"; JMS SBRT;
FNOP;

```


0567	1603	06	643	RDO; CLC; ADD RE; RAR; JCN ZCY FINDCPA;
0568	1609	06	649	REM PAST-CPA INDICATED BY ALL F'S OUTPUT;
0569	1614	06	649	LCM "F"; FIM RC "EO"; FLCCP: SRC RO; WRM;
0570	1616	06	649	ISZ R1 FLCCP;
0571	1622	06	650	FIM R4 "00"; FIM R6 "AC"; JMS TRANSLCCP;
0572	1622	06	650	REM IF PAST-CPA COMPUTE TARGET
0573	1622	06	656	COURSE AND SPEED;
0574	1622	06	656	JUN FINDCANDS;
0575	1624	06	656	REM ELSE COMPUTE CPA BEARING, RANGE AND TIME;
0576	1624	06	658	FIMCCPA: FIM R4 "10"; FIM R6 "60"; JMS TRANSLCCP;
0577	1624	06	658	FIM R4 "20"; FIM R6 "70"; JMS TRANSLCCP;
0578	1630	06	658	FIM R4 "00"; FIM R6 "AC"; JMS TRANSLCCP;
0579	1636	06	65E	JMS CORDICAT;
0580	1642	06	664	REM POSITICNS RCTATED, RCTATE CPA BEARING
0581	1644	06	66A	IF MINUS 90 DEGREES;
0582	1644	06	66C	FIM RA "EO"; SRC RA; RCT3; JCN ZAC NOBACKROTATE1;
0583	1644	06	66C	FIM R4 "30"; FIM R0 "MINP12; JMS FERT;
0584	1644	06	66C	FIM R4 "00"; FIM R2 "3C"; JMS ADRT;
0585	1650	06	672	SRC R0 "00"; RAR; JCN ZCY NOBACKROTATE1;
0586	1656	06	678	FIM R4 "30"; FIM R0 TWCPI; JMS FERT;
0587	1662	06	67E	FIM R4 "00"; FIM R2 "3C"; JMS ADRT;
0588	1667	06	683	REM PUT VALUES INTO CLTPLT FCR CPA BEARING
0589	1673	06	689	AND RANGE;
0590	1679	06	68F	NOBACKROTATE1;
0591	1679	06	68F	FIM RA "08"; FIM RC "E6";
0592	1679	06	68F	LDM 12; XCH RF; JMS TRANSPART;
0593	1683	06	693	FIM RA "29"; FIM RC "EC"; LDM 10; XCH RF;
0594	1687	06	697	JMS TRANSPART;
0595	1693	06	69D	FIM RC "E5"; LCM 11; XCH RB; JMS FBACK;
0596	1695	06	69F	FIM R4 "60"; FIM R6 "10"; JMS TRANSLCCP;
0597	1701	06	6A5	FIM R4 "70"; FIM R6 "20"; JMS TRANSLCCP;
0598	1707	06	6AB	FIM R0 "10"; FIM R2 "8C"; JMS SBRT;
0599	1713	06	6B1	FIM R0 "20"; FIM R2 "9C"; JMS SBRT;
0600	1719	06	6B7	JMS CORDICAT;
0601	1727	06	6BD	FIM R6 "CC"; FIM R8 "00";
0602	1727	06	6BF	FIM R4 "20"; JMS DVRT;
0603	1733	06	6C5	FIM R2 "10"; JMS ACRT;
0604	1737	06	6C9	FIM R0 "B0";
0605	1743	06	6CF	REM INVERT CPA TIME TO PROCFER HRS AND MINS AND STORE OUT;
0606	1743	06	6CF	CONVERT CPA TIME TO PROCFER HRS AND MINS AND STORE OUT;
0607	1743	06	6CF	FIM R4 "B0"; FIM R6 "CC"; JMS TRANSLCCP;
0608	1743	06	6CF	FIM R4 "10"; FIM R0 DAY; JMS FERT;
0609	1743	06	6D5	

[illegible]


```

0661 1581 07 78D
0664 1581 07 78D
0665 1581 07 78D
0666 1581 07 78D
0667 1581 07 78D
0668 1581 07 78D
0669 1581 07 78D
0670 1581 07 78D
0671 1581 07 78D
0672 1581 07 78D
0673 1581 07 78D
0674 1581 07 78D
0675 1581 07 78D
0676 1581 07 78D
0677 1581 07 78D
0678 1581 07 78D
0679 1581 07 78D
0680 1581 07 78D
0681 1581 07 78D
0682 1581 07 78D
0683 1581 07 78D
0684 1581 07 78D
0685 1581 07 78D
0686 1581 07 78D
0687 1581 07 78D
0688 1581 07 78D
0689 1581 07 78D
0690 1581 07 78D
0691 1581 07 78D
0692 1581 07 78D
0693 1581 07 78D
0694 1581 07 78D
0695 1581 07 78D
0696 1581 07 78D
0697 1581 07 78D
0698 1581 07 78D
0699 1581 07 78D
0700 1581 07 78D
0701 1581 07 78D
0702 1581 07 78D
0703 1581 07 78D
0704 1581 07 78D
0705 1581 07 78D
0706 1581 07 78D
0707 1581 07 78D
0708 1581 07 78D
0709 1581 07 78D
0710 1581 07 78D

```

```

REM OUTPUT TARGET COURSE AND SPEED;
FIM RC "F0"; LDM 1; XCH RB; JMS DCRAM;
FIM RA "08"; FIM RC "F6"; LDM 12; XCH RF;
JMS TRANS PART;
FIM RC "10"; JMS ZERORAM; RC; WRM;
LDM 5; FIM RC "15"; SRC RC; FIM R6 "30"; JMS MLRT;
FIM R2 "20"; FIM R4 "1C"; FIM R6 "30"; XCH RF;
FIM RA "38"; FIM RC "FC"; LDM 13; XCH RF;
JMS TRANS PART;
FIM RC "F2"; LDM 14; XCH RB; JMS FBACK;
ATLASTOHLAST: JUN CPARETURN;
REM END MANEUVERING BCARC ALGORITHM;

REM ROTATES POSITIONS PLUS 90 DEGREES;
REVOLVE:
FIM R4 "10"; FIM R6 "2C"; JMS TRANSLCCP;
FIM R4 "20"; FIM R6 "10"; JMS TRANSLOOP;
FIM R4 "30"; FIM R6 "2C"; JMS TRANSLOOP;
FIM R4 "70"; JMS CPLRT;
FIM R4 "70"; JMS CPLRT;
FIM R4 "80"; FIM R6 "6C"; JMS TRANSLCCP;
FIM R4 "90"; FIM R6 "3C"; JMS TRANSLCCP;
FIM R4 "80"; FIM R6 "80"; JMS TRANSLCCP;
FIM R4 "30"; JMS CPLRT;
FIM R4 "30"; FIM R6 "90"; JMS TRANSLOOP;
LDM "F"; FIM R0 "E0"; SRC R0; WR3;
BBL 0;

REM ZEROES A RAM REGISTER AND ITS SIGN;
ZERGRAM: CLB;
ZRAM: SRC RC; WRM; ISZ RC ZRAM; WRO; BBL 0;

REM TRANSFERS BEARING FROM STORAGE TO
WORKING REGISTER;
LOADBRG:
JMS ZERORAM; LDM 8; XCH RD;
LDM 12; XCH RF; JMS TRANS PART; BBL 0;

REM TRANSFERS RANGE FROM STORAGE TO
WORKING REGISTER;
LOADBRG:
JMS ZERORAM; LDM 9; XCH RD;
LDM 10; XCH RF; JMS TRANS PART; BBL 0;

REM TRANSFERS PART OF CNE REGISTER TO ANOTHER;
TRANS PART: SRC RA; RCM; SRC RC; WRM;
INC RB; INC RD; ISZ RF TRANS PART; BBL 0;

```


0807 2351 09 957
 0808 2239 09 95D
 0809 2403 09 963
 0810 2407 09 967
 0811 2411 09 968
 0812 2414 09 96E
 0813 2421 09 975
 0814 2425 09 97S
 0815 2427 09 97B
 0816 2429 09 97D
 0817 2431 09 97F
 0818 2433 09 981
 0819 2433 09 981
 0820 2433 09 981
 0821 2434 09 982
 0822 2434 09 982
 0823 2434 09 982
 0824 2434 09 982
 0825 2440 09 988
 0826 2442 09 98A
 0827 2445 09 991
 0828 2449 09 991
 0829 2451 09 993
 0830 2458 09 99A
 0831 2458 09 99A
 0832 2458 09 99A
 0833 2459 09 99B
 0834 2466 09 9A2
 0835 2467 09 9A3
 0836 2469 09 9A5
 0837 2477 09 9AD
 0838 2477 09 9AD
 0839 2477 09 9AD
 0840 2477 09 9AD
 0841 2477 09 9AD
 0842 2480 09 9B0
 0843 2486 09 9B6
 0844 2488 09 9B8
 0845 2494 09 9BE
 0846 2502 09 9C6
 0847 2502 09 9C6
 0848 2502 09 9C6
 0849 2502 09 9C6
 0850 2509 09 9CD
 0851 2514 09 9D2

SKIPF2: FIM R2 "00"; FIM R2 "30"; JMS SBRT;
 FIM R4 "20"; FIM R6 "10"; JMS TRANSLCCP;
 FIM R0 "10"; JMS CPLRT;
 REM DO REST 2; CF XCH RC; LCM 12; XCH RA;
 JOIN2: JMS SETUP; XCH RC; RDO; RAR; JCN ZCY SKIPX;
 MLOOP: FIM R0 "00"; JCN JCINX;
 LCM "F"; XCH O; XCH RD;
 SKIPX: JMS RCTATE;
 JOINX: JMS RCTATE;
 ISZ RA MLOOP;
 REM END CORDIC ROTATION MODE;
 ATLAS: BBL O;
 REM COMPLEMENTS REGISTER;
 CPLRT: COMPL: SRC R0; LCM 6; ADM; CMA; WRM;
 ISZ R1 CCML; LCM 0; SRC RC; ACM; DAA; WRM; INC R1;
 STC; TENS; TENS;
 JCN NZCY TENS;
 SRC R0; RDO; RAR; CMC; RAL; WRO; EBL G;
 REM ADDS TWO REGISTERS INTO FIRST;
 ADRT: CLC;
 ADDR: SRC R0; RDM; SRC R2; ADM; DAA; SRC R0; WRM;
 INC R3;
 ISZ R1 ADDR;
 RDO; XCH RF; SRC R2; RCC; ADD RF; SRC R0; WRO; BBL O;
 REM SUBTRACTS TWO REGISTERS, ANSWER IN FIRST;
 SBRT: LCM 10; XCH RF; CLB;
 LDM 10; SRC R0; RDM; SRC R2; SBM; JCN NZCY OK;
 SUBT: RF; CLC;
 ADD RF; CLC;
 OK: CMC; SRC RC; WRM; INC R3; ISZ R1 SUBT;
 RDO; XCH RF; SRC R2; RDO; ADD RF; SRC R0; WRO; BBL O;
 REM TRANSFERS CCNTENTS CF CNE REGISTER TO ANOTHER;
 TRANSLCCP:
 SRC R4; RDM; SRC R6; WRM; INC R5; ISZ R7 TRANSLOOP;
 SRC R4; RDO; SRC R6; WRO; EBL O;


```

0856 514 09 5D2
0857 514 10 A00
0858 560 10 A00
0859 560 10 A00
0860 560 10 A00
0861 560 10 A00
0862 560 10 A00
0863 560 10 A00
0864 560 10 A00
0865 560 10 A00
0866 560 10 A00
0867 560 10 A00
0868 560 10 A00
0869 560 10 A00
0870 560 10 A00
0871 560 10 A00
0872 560 10 A00
0873 560 10 A00
0874 560 10 A00
0875 560 10 A00
0876 560 10 A00
0877 560 10 A00
0878 560 10 A00
0879 560 10 A00
0880 560 10 A00
0881 560 10 A00
0882 560 10 A00
0883 560 10 A00
0884 560 10 A00
0885 560 10 A00
0886 560 10 A00
0887 560 10 A00
0888 560 10 A00
0889 560 10 A00
0890 560 10 A00
0891 560 10 A00
0892 560 10 A00
0893 560 10 A00
0894 560 10 A00
0895 560 10 A00
0896 560 10 A00
0897 560 10 A00
0898 560 10 A00
0899 560 10 A00
0900 560 10 A00
0901 560 10 A00
0902 560 10 A00

```

```

BEGIN 2560;
REM CORDIC VECTORING MCDE FOR ARCTAN AND
CORDIC FUNCTIONS;
CORDIC:
REM ZERO RAM FOR Z IN "0C", X IN "10", Y IN "20";
SYN A=2,AP=3,B=4,BP=5,C=6,CP=7,CEASE=14;
FIM RO "00"; CLB;
ZEROCLOOPAT: SRC RO; WRN; ISZ R1 ZEROLCOPAT; WRO;
FIM R4 "20"; SRC R6 "2C"; FIM R6 "2C"; JMS TRANSLCOPAT; WRO;
SRC RO; RCO; RAR; JCN ZCY SWITCH1;
FIM R4 "30"; JMS CPLRT;
SWITCH1: FIM R4 "10"; FIM R6 "40"; JMS TRANSLCOPAT;
SRC R6; RCO; RAR; JCN ZCY SWITCH2;
FIM RO "40"; JMS CPLRT;
FIM TEST Y/X: JMS MINIMIZE TRUNCATION ERROR;
SWITCH2: FIM R4 "30"; FIM R2 "4C"; JMS SBRT;
SRC R4 "00"; RAR; RCO MINPI; NZCY SWITCH3;
FIM R4 "10"; FIM R6 "30"; JMS TRANSLCOPAT;
FIM R4 "20"; JMS CPLRT;
FIM R4 "30"; JMS TRANSLCOPAT;
REM ENSURE X RC "10"; SRC RO; RCO; RAR;
SWITCH3: SWITCH4: RO "2C"; JMS CPLRT;
JCN ZCY SWITCH4; FIM R4 "30"; FIM R2 "30"; JMS SBRT;
FIM R4 "00"; FIM R6 "20"; JMS TRANSLCOPAT;
REM TEST Y AND DRIVE TC C;
SWITCH4: JMS RC; LDM C;
MLOOPAT: FIM R4 "20"; SRC RO; RCO; RAR;
JCN NZCY: SKIPXAT; JUN JCINXAT;
LDM "F"; XCH RD;
SKIPXAT: LDM C; XCH RD;
JCINXAT: JMS RCTATE;
ISZ RA MLOOPAT;
REM X=X#1/K;
CHECK:
FIM R4 "30"; FIM RO INVK; JMS FERT; JMS MLRT;
FIM B "30"; FIM A "10"; FIM C "20"; JMS MLRT;
FIM RO "00"; SRC RO; RCO; RAR; JCN ZCY ATLASTAT;
FIM R4 "30"; FIM RO "30"; JMS ACRT;
FIM RO "00";

```



```

LDM O;SRC B;WR2;
MML7:BBL O;

SYN A,AP,B,BP,C,CP,CBASE;

BEGIN 3072;

SYN A=2,AP=3,CNT=3,R=4,RP=5,C=6,CP=7,
C=8,CP=9,
NN=10,NP=11,K=12,KP=13,SHIFTA=14, SHIFTC=15;

MADRT:CLC;
STMD:SRC R2;RDM;SRC R6;ADM;DAA;WRM;
ISZ R3 SKIPM1;EBL O;
SKIPM1: ISZ R7 STMD; EBL O;

REM DECIMAL CIVISION ROUTINE,
WRITTEN BY
G. A. KILLDALL,
ASSISTANT PROFESSOR, SCHCGL,
NAVAL POSTGRADUATE, CALIFORNIA,
MONTEREY,
DECDIV: LDM 9; SRC A; WR2; SRC C; WR2; SRC Q; WR2;
CLB; ZERRC: SRC G; WRM; SRC R; WRM; INC RP;
ISZ QP ZERRC; LCCATEZERC: LC NP; CMA; XCH AP; SRC A;
CLB; XCH NP;
RDM; NZAC FOUNDZERO; ISZ NP LOCATEZERO;
JUN ENDDIVIDE; NP; XCH RP; CLB; XCH AP;
FOUNDZERO: LD RCM; SRC R; WRM; INC AP;
COPYA: SRC A;
ISZ RP COPYA;
LD NP; XCH SHIFTA; SRC A; RD2; ADD NP; XCH NP; TCC;
XCH N;
CLB; XCH KP;
LOCATEZERO1: LC KP; CMA; XCH CP; SRC C; RDM;
JCN NZAC FOUNDZERO1; ISZ KP LOCATEZERO1; EBL 1;

FOUNDZERO1: LC KP; XCH SHIFTC; RD2; ADD KP; XCH KP;
TCC; XCH K;
SRC Q; RD2;
CLC; Q; RD2;
CLC; LD KP;
JCN NZCY UNDERF; BBL C;
UNDERF: JCN NZAC DOVERFLCW;

LDM 15; XCH NP; LD C; XCH N;
COPYC1: SRC C; RDM; SRC N; WRM;
LD CP; JCN ZAC PASTCOPY1; DAC; XCH CP;

```


1047 3183 12
 1048 3188 12
 1049 3188 12
 1050 3191 12
 1051 3196 12
 1052 3198 12
 1053 3200 12
 1054 3202 12
 1055 3207 12
 1056 3211 12
 1057 3214 12
 1058 3218 12
 1059 3222 12
 1060 3228 12
 1061 3230 12
 1062 3233 12
 1063 3237 12
 1064 3244 12
 1065 3246 12
 1066 3253 12
 1067 3255 12
 1068 3259 12
 1069 3263 12
 1070 3267 12
 1071 3272 12
 1072 3277 12
 1073 3277 12
 1074 3282 12
 1075 3283 12
 1076 3284 12
 1077 3284 12
 1078 ERRORS.
 0

LD NP; DAC; XCH NP; JUN CPGYCL;
 PASTCOPY1: LD NP; JCN ZAC CIV;
 DAC; XCH NP; SRC N; LDM C; WRM;
 JUN; PASTCOPY1;
 DIV: LDM 10; XCH K;
 SUBO: CLB; XCH CNT;
 SUB1: CLB; XCH RP; LD NP; XCH CP; SRC R;
 SUB2: RDM; SRC C; SBM; JCN NZCY COMPLEMENT;
 ADD K; CLC;
 CCOMPLEMENT: CMC; SRC R; WRM; INC RP;
 SRC R; ISZ SUB2;
 LD RP; JCN ZAC CHECKCARRY;
 RDM; SUB CP; WRM; CMC;
 CHECKCARRY: JCN NZCY CARRYOUT;
 INC CNT; JUN SUB1;
 CARRYOUT: LD NP; XCH CF; CLB; XCH RP; INC RP;
 ADD4: SRC C; RDM; SRC R; ADM; DAA; WRM;
 ISZ CP ADD4; ZAC SKIPADD; TCC; SRC R; ADM; WRM;
 LD RP; JCN ZAC G; LD CNT; WRM; LD CP; JCN ZAC ENDDIVIDE;
 SKIPADD: SRC C; ISZ NP SUBO;
 DAC; XCH QP;
 ENDDIVIDE: CLB; XCH NP; LD SHIFTC; XCH CP;
 CPGYC2: SRC C; RDM; SRC N; WRM; INC NP;
 ISZ CP CPGYC2;
 LD NP; JCN ZAC PASTFILL;
 FILLZ: SRC N; CLB; WRM; ISZ NP FILLZ;
 PASTFILL: BBL 0;
 DOVERFLOW: BBL 1;
 SYN A,AP,CNT,R,RP,C,CP,G,CP,N,NP,K,KP,SHIFTA,SHIFTC;
 END

LIST OF REFERENCES

1. Tinklepaugh, N., and Roth, A. "CPA Calculator." System description prepared at Naval Electronics Laboratory Center, San Diego, Ca., 1972. (Photocopied.)
2. Tinklepaugh, N., and Roth, A. "Closest Point of Approach Calculator." Demonstration materials prepared at Naval Electronics Laboratory Center, San Diego, Ca., 1972. (Mimeographed.)
3. Fritts, D. E. "Device Description, Scientific Read Only Memory (P/N 15171NA)." Specification prepared at North American Rockwell Microelectronics Company, Anaheim, Ca., October 25, 1971. (Photocopied.)
4. North American Rockwell Microelectronics Company. "Functional Description: PCPU Sample Calculator." Product bulletin, Anaheim, Ca., October, 1971. (Photocopied.)
5. ~~and A. Roth~~. Telephone interview with N. Tinklepaugh and ~~A. Roth~~, Naval Electronics Laboratory Center, San Diego, Ca. April 5, 1973.
6. Dutton, Benjamin. Navigation and Nautical Astronomy. 9th ed. revised. ~~Annapolis: United States Naval~~ Institute, 1948.
7. Kildall, Gary A. "A Chebyshev Approximation Routine for the Intel MCS-4 Computer." Monterey, Ca.: Computer Science Group, United States Naval Postgraduate School, September, 1972. (Mimeographed.)
8. Volder, Jack E. "The CORDIC Trigonometric Computing Technique," IRE Transactions on Electronic Computers, Vol. EC-8, No. 3 (September, 1959), pp. 330-334.
9. Walther, J. S. "A Unified Algorithm for Elementary Functions," Spring Joint Computer Conference, Vol. 38 (May, 1971), pp. 379-385.
10. Schmid, Herman, and Bogacki, Anthony. "Use decimal CORDIC for generation of many transcendental functions," EDN, Vol. 18, No. 4 (February 20, 1973), pp. 64-73.
11. Walther, J. S. Letter to Dr. Richard Franke, United States Naval Postgraduate School, Monterey, Ca., from Hewlett-Packard Laboratories, Palo Alto, Ca., February 13, 1973.
12. Kildall, Gary A. "An MCS-4 Assembler and Interpreter for the IBM System/360." Monterey, Ca.: Computer Science Group, United States Naval Postgraduate School, July, 1972. (Mimeographed.)
13. Intel Corporation. MCS-4 Micro Computer Set: Users Manual. Rev. 3. Santa Clara, Ca.: Intel Corporation, July, 1972.
14. Brubaker, Ray H., Jr. "I/O Simulation using the MCS-4 Interpreter for the IBM System/360." Monterey, Ca.: Computer Science Group, United States Naval Postgraduate School, July, 1972. (Mimeographed.)

INITIAL DISTRIBUTION LIST

	No Copies
1. Defense Documentation Center Cameron Station Alexandria, Virginia 23314	2
2. Library, Code 0212 Naval Postgraduate School Monterey, California 93940	2
3. Chairman, Computer Science Group, Code 72 Naval Postgraduate School Monterey, California 93940	1
4. Asst Professor G. A. Kildall, Code 53 Kd Naval Postgraduate School Monterey, California 93940	1
5. Inst Ray H. Brubaker, Jr., Code 53 Bh Naval Postgraduate School Monterey, California 93940	1
6. Asst Professor R. Franke, Code 53 Fe Naval Postgraduate School Monterey, California 93940	1
7. Mr. Norm Tinklepaugh, Code 4300 Naval electronics laboratory Center San Diego, California 92152	1
8. LCDR K. H. Kerns 128 Belle Drive Marina, California 93933	1
9. LT R. S. Cooper 2200 South Ocean Boulevard Apartment 608 Delray Beach, Florida 33444	1

DOCUMENT CONTROL DATA - R & D

(Security classification of title, body of abstract and indexing annotation must be entered when the overall report is classified)

1. ORIGINATING ACTIVITY (Corporate author)		2a. REPORT SECURITY CLASSIFICATION	
Naval Postgraduate School Monterey, California 93940		Unclassified	
3. REPORT TITLE		2b. GROUP	
A MICROCOMPUTER SOLUTION TO MANEUVERING BOARD PROBLEMS			
4. DESCRIPTIVE NOTES (Type of report and inclusive dates)			
Master's Thesis; June 1973			
5. AUTHOR(S) (First name, middle initial, last name)			
Kenneth Harper Kerns Roger Stuart Cooper			
6. REPORT DATE		7a. TOTAL NO. OF PAGES	7b. NO. OF REFS
June 1973		83	14
8a. CONTRACT OR GRANT NO.		9a. ORIGINATOR'S REPORT NUMBER(S)	
b. PROJECT NO.			
c.		9b. OTHER REPORT NO(S) (Any other numbers that may be assigned this report)	
d.			
10. DISTRIBUTION STATEMENT			
Approved for public release; distribution unlimited.			
11. SUPPLEMENTARY NOTES		12. SPONSORING MILITARY ACTIVITY	
		Naval Postgraduate School Monterey, California 93940	
13. ABSTRACT			

A special-purpose digital computer has been developed using the new MOS LSI microprocessor technology. The primary goal of this work was to solve a fairly complicated task using a minimal amount of random logic and limited development time. This computing system solves ships' maneuvering board problems including the determination of course, speed, and closest point of approach of other ships. Ten contacts can be tracked simultaneously. The triangulation involved in these calculations is performed using a decimal version of the CORDIC algorithm. A complete maneuvering situation can be computed in 4.5 seconds. Except for hardware required to drive the displays, all logic is contained in the software-encoded read-only-memory chips, which drive a single CPU chip. Approximately eight hundred man-hours were required to develop the programs and prototype the hardware. The system is modular and easy to maintain, low in bulk, power consumption, and cost.

14. KEY WORDS	LINK A		LINK B		LINK C	
	ROLE	WT	ROLE	WT	ROLE	WT
CORDIC algorithm						
CPA (closest point of approach)						
LED (light-emitting diode) display						
maneuvering board						
MCS-4 Micro Computer Set						
microcomputer						
microprocessor						
PROM (programmable read-only-memory)						
RAM (random-access-memory)						
relative movement problem						
ROM (read-only-memory)						

Thesis

K3875

Kerns

c.1

A microcomputer solution to maneuvering board problems.

145316

15 MAR 74

21335

7 JUL 77

24656

13 OCT 78

24676

20 OCT 78

25293

12 MAR 79

25203

11 FEB 81

26393

6 SEP 88

33568

Thesis

K3875

Kerns

c.1

A microcomputer solution to maneuvering board problems.

145316

thesK3875

A microcomputer solution to maneuvering



3 2768 002 12128 7

DUDLEY KNOX LIBRARY